We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

14

A Robust and Flexible Control System to Reduce Environmental Effects of Thermal Power Plants

Toru Eguchi, Takaaki Sekiai, Naohiro Kusumi, Akihiro Yamada,
Satoru Shimizu and Masayuki Fukai
Hitachi Ltd.
Japan

1. Introduction

Regulations on environmental effects due to such issues as nitrogen oxide (NOx) and carbon monoxide (CO) emissions from thermal power plants have become stricter[1]; hence the need for compliance with these regulations has been increasing. To meet this need, several technologies with respect to fuel combustion, exhaust gas treatment and operational control have been developed[2-4]. The technologies for the fuel combustion and the exhaust gas treatment include a low NOx burner and an air quality control system, and they are capable of reducing impact on the environment as physical and chemical implementation methods. The operational control technology for the thermal power plants is constantly required to receive changes in operational conditions. It is difficult to realize operational control which responds to combustion properties.

To overcome this issue, the operational control must be able to reduce NOx and CO emissions flexibly in accordance with such changes. Robustness is also required in such control because the measured NOx and CO data often include noise. Therefore, a robust and flexible plant control system is strongly desired to reduce environmental effects from thermal power plants efficiently.

Several studies have proposed plant control technologies to reduce the environmental effects[4-10]. These technologies are classified into two types of methods: model based and non-model based methods. The former methods include an optimization algorithm and a numerical model to estimate plant properties using neural networks (NNs)[11,12] and multivariable model predictive control[13]. The optimization algorithm searches for optimal control signals to reduce NOx and CO emissions using the numerical model. The latter methods have no models and they generates the optimal control signals by fuzzy logic[14]. A fuzzy logic controller outputs the optimal control signals for multivariable inputs using fuzzy rule bases. The fuzzy rule bases are based on *a priori* knowledge of plant control, and they can be tuned by parameters.

These technologies require the measured plant data for initial tuning of the model properties and the parameters of rules when the technologies are installed in plants. It usually takes some time to collect enough plant data. In addition, the search for control

signals can only be made in the past operating range, thus it is difficult to find the optimal control signals of they are located outside the range.

The authors have proposed a new plant control system for reducing environmental effects utilizing numerical calculation technology[15] to shorten the time for initial tuning and search the global optima. The system has one or more calculation databases (DBs) with respect to NOx and CO properties obtained by numerical calculation. Since the model can be tuned using the calculation DBs in advance, it is not necessary to take times for initial tuning when the control system is installed. Moreover, the proposed system obtains better control signals than the conventional technologies because it can model the NOx and CO properties including both inside and outside the operating range by the numerical calculation, which facilitates to search the optimal control signals.

After installation, the proposed control system is capable of tuning its model using the data measured in real time to reduce the model errors. In plant control, the shortest interval for changing operations is every 20 minutes because it often takes about 20 minutes to become static after an operation. The proposed system must be able to calculate the control signals during this interval, hence model tuning and searching for control signals should terminate within 20 minutes.

The proposed system employs radial basis function (RBF) network[16,17] and reinforcement learning (RL)[18]. The RBF network represents the NOx and CO properties to estimate their concentrations according to the control signals. The RL leads to the optimal control signals to achieve the control goals which is to reduce the estimated NOx and CO concentrations. The RBF network is one of the NNs having Gaussian basis functions. The RBF network usually learns the NOx and CO properties faster than ordinary NNs because the learning algorithm of the RBF network can be converted into matrix calculations without iterations. The RL is one of the machine learning methods[19] optimizing action rules of an agent by trial and error. It is preferable to apply the RL to the control system which requires real-time computing because the RL is a single point searching method and its computational cost is relatively small. It is also preferable to use the RL because the control history which can be utilized to improve the control logic can be traced in the RL control system. The proposed control system with the above features is expected to realize robustness, flexibility in control and real-time computing.

However, there are two practical problems to enhance these advantages more efficiently in the proposed control system. The first one is ensuring that the model can achieve enough estimation accuracy within practical computational times. Conventional methods to improve estimation accuracy of the model[11] are to adjust radii parameters of the Gaussian basis functions in RBF networks by calculating the estimation error for regression. However, with the conventional radius adjustment methods it might be difficult to adjust radii parameters within the time restriction because the adjustment of radii by regression requires many iterations. On the other hand, a radius adjustment method without calculation of estimation error has also been proposed[20]. This method determines the radii parameters using an equation considering learning data properties such as size and dimension. Its computational time is fast, but the estimation accuracy is worse than the method with regression. Therefore, it is desired to propose a new radius adjustment method for the plant control to achieve both higher estimation accuracy and faster computation.

The authors propose a novel radius adjustment method to overcome this first problem[21]. The proposed method focuses on the importance of covering input space properly where the model simulates the NOx and CO properties by the Gaussian basis functions to improve

estimation accuracy. This method adjusts radii parameters considering distances among the learning data. Consequently, the Gaussian basis functions can cover the input space properly and both high estimation accuracy and practical computational speed are achieved.

The second problem is to improve flexibility of the learning algorithm. Performance of the RL depends on the definition of a reward function equivalent to an evaluation function. The reward function has to be defined so that the RL algorithm can obtain the desired goal for the problem. As for application of the RL to thermal power plant control, the properties of the model changes in accordance with operational changes, thus the reward function has to be changed flexibly for the operational changes. However, it is quite difficult to prepare the reward functions for all patterns of operational conditions in advance.

To overcome this second problem, the authors introduce a reward function which has variable parameters and they proposed an automatic reward adjustment method[22]. The proposed method adjusts the variable parameters of the reward function automatically based on the NOx and CO emissions obtained in the learning process. As a result, the proposed method can obtain proper reward functions for all kinds of operational conditions.

The following sections outline the proposed control system and its newly proposed methods. Simulations clarify the advantages of the proposed system with respect to the following points: estimation accuracy and computational time of the RBF network, flexibility of the control logic and robustness in control for the noise of data.

2. Proposed plant control system for reducing environmental effects

2.1 Basic structure

Figure 1 shows the basic structure of the proposed control system. This system consists of a plant property estimation part and an operation optimizing part. The plant property estimation part includes a statistical model and measurement and numerical calculation DBs. The statistical model estimates the NOx and CO emission properties in thermal power plants. It is difficult to express these properties as mathematical equations because they have strong nonlinearities. The proposed system employs the RBF network as the statistical model which can estimate NOx and CO emissions for control variables using data stored in the DBs. The measurement DBs store the measured NOx and CO data for some control variables, and the numerical calculation DB stores data consisting of NOx and CO values for control variables calculated by the combustion analysis[15]. The control variables correspond to input of the statistical model, and the estimated NOx and CO emissions correspond to output of it. The statistical model can be modified by measured data obtained during the plant operations.

Conventional studies have been made about the model based control technology to reduce environmental effects from thermal power plants[4,6-8], but none of them considered employing not only the measured DB, but also the numerical calculation DBs. As the model can be tuned using the calculation DBs in advance, it is not necessary to take times for initial tuning at the time of installation. In addition, it is possible to tune the model after the installation by the data in the measured DB.

The operation optimizing part includes a RL agent, a reward calculation module, a reward adjustment module and a learning result DB. The learning procedure is as follows. First, the statistical model calculates and outputs the model outputs for the model inputs changed by

the RL agent. Secondly, the reward calculation module calculates a reward using the model inputs and gives it to the RL agent. Thirdly, the RL agent learns its control logic. Learning results are stored in the learning result DB, and they are converted into modification signals. The control signals of the plant are generated by adding the modification signals to original control signals of the basic controller. The reward adjustment module adjusts reward parameters using the model outputs and the calculated reward. Normalized Gaussian network (NGnet)[23] has been employed as the structure of the RL agent. The learning algorithm of the NGnet is an actor-critic learning method[18], and it is appropriate for learning in a continuous environment.

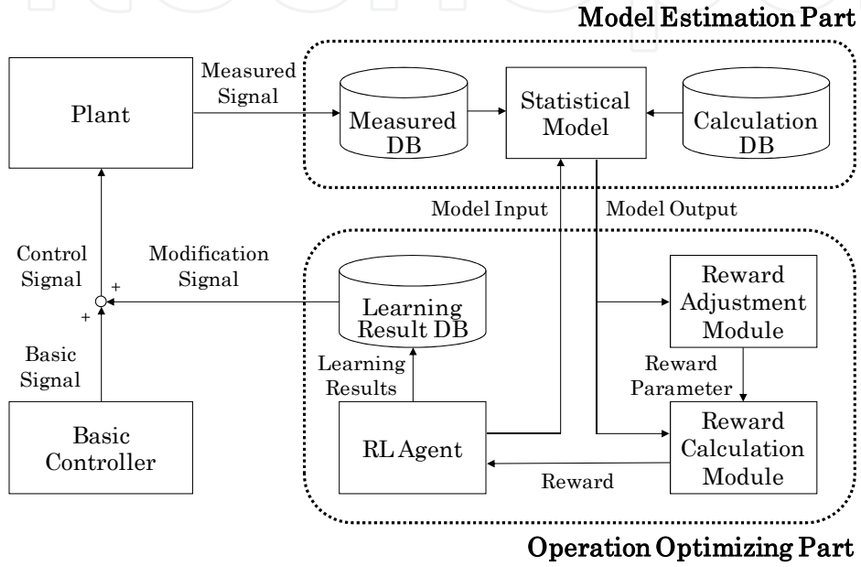


Fig. 1. Basic Structure of the Proposed Plant Control System

2.2 RBF network

The basic structure of the RBF network is shown in Fig. 2. The RBF network has three layers: an input layer, a hidden layer with Gaussian function, and an output layer. First, the J-dimensional vector is input in the input layer. Secondly, Gaussian function values are calculated using the input in the hidden layer. Finally, the P-dimensional vector is calculated by the Gaussian function values and weight parameters in the output layer. The RBF network is preferred for constructing a response surface due to the following properties.

- The RBF network avoids overfitting by the parameter of weight decay[16] to reduce the influences of noise included in the learning data.
- The RBF network does not need iterative calculations for learning of weight parameters like back propagation does[12].

Here, the input and output of the RBF network are denoted as $\mathbf{x}^T = \{x_1, \ldots x_j, \ldots x_J\}$ $(j \in J)$, $\mathbf{y}^T = \{y_1, \ldots y_p, \ldots y_P\}$ $(p \in P)$, then the p-th output y_p is calculated by Eqs. (1) and (2).

$$h_l(\mathbf{x}) = \exp\left(-\frac{(\mathbf{x} - \mathbf{c}_l)^T (\mathbf{x} - \mathbf{c}_l)}{r_l^2} \right) \tag{1}$$

$$y_p(\mathbf{x}) = \sum_{l=1}^{N_M} u_{lp} h_l(\mathbf{x}) \tag{2}$$

Here, $h_l(\mathbf{x})$ is the Gaussian function value of the l-th basis function, N_M is the number of basis functions, u_{lp} is the weight parameter between the hidden layer and output layer and $\mathbf{c}_l, r_l$ are center coordinates and radius of the l-th basis function, respectively. The parameters $\mathbf{c}_l$ and r_l should be determined appropriately because they have much influence on estimation accuracy. In this chapter, the center coordinates are set to the learning data, and the radii are adjusted by the proposed radius adjusting method described later.

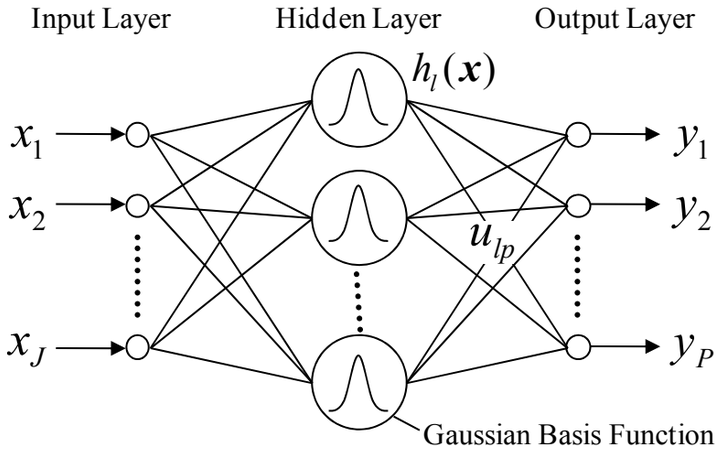


Fig. 2. Basic Structure of RBF Network

Learning of the RBF network corresponds to the determination of the weight parameter u_{lp} to minimize the energy function E_p given by Eq. (3) when the teaching data paired with learning data $\mathbf{x}_q$ are denoted as y_{pq}.

$$E_p = \sum_{q=1}^{N_D} (y_{pq} - y_p(\mathbf{x}))^2 + \lambda \sum_{l=1}^{N_M} u_{lp}^2 \tag{3}$$

Here, N_D is the number of learning data and λ is a weight decay reducing influences of noise included in learning data. The proposed control system can realize a robust control by tuning this parameter in accordance with the learning data. Then, the following matrices are defined.

$$\mathbf{U} = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1P} \\ u_{21} & u_{22} & \cdots & u_{2P} \\ \vdots & \vdots & \ddots & \vdots \\ u_{N_M 1} & u_{N_M 1} & \cdots & u_{N_M P} \end{bmatrix} \tag{4}$$

$$\mathbf{H}=\begin{bmatrix} h_1(\mathbf{x}_1) & h_2(\mathbf{x}_1) & \cdots & h_{N_M}(\mathbf{x}_1) \\ h_1(\mathbf{x}_2) & h_2(\mathbf{x}_2) & \cdots & h_{N_M}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ h_1(\mathbf{x}_{N_D}) & h_2(\mathbf{x}_{N_D}) & \cdots & h_{N_M}(\mathbf{x}_{N_D}) \end{bmatrix} \tag{5}$$

$$\mathbf{Y}=\begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1P} \\ y_{21} & y_{22} & \cdots & y_{2P} \\ \vdots & \vdots & \ddots & \vdots \\ y_{N_M 1} & y_{N_M 1} & \cdots & y_{N_M P} \end{bmatrix} \tag{6}$$

$$\mathbf{\Lambda}=\lambda\mathbf{I} \tag{7}$$

In Eq. (3), both sides are partially differentiated by u_{lp} and Eqs. (4)-(7) are substituted, then Eq. (8) is obtained[16]. The learning of the RBF network can be described as the calculation of the weight matrix $\mathbf{U}$ given by Eq. (8).

$$\mathbf{U}=\left(\mathbf{H}^T\mathbf{H}+\mathbf{\Lambda}\right)^{-1}\mathbf{H}^T\mathbf{Y} \tag{8}$$

2.3 Reinforcement learning

2.3.1 Basic algorithm

The NGnet for learning of the RL agent learns its action, *i.e.*, control logic, and state value by putting Gaussian basis functions on its state space. Here, the state space is a mapping space to identify its status in the learning environment. The state value is a degree to evaluate how desirable the agent is in its current state. NGnet is known to be able to learn faster than other RL algorithms such as tile coding[18] because of the following features.

- NGnet can learn locally by the Gaussian basis functions.
- NGnet can reduce necessary basis function size by normalization.
- NGnet can add/delete the basis functions and parameter tuning.

Figure 3 shows the basic structure of NGnet. First, NGnet calculates activations of its Gaussian basis functions a_i and normalized activations b_i for the input $\mathbf{x}$ by Eqs. (9)-(11). Next, outputs of actor $\mathbf{m}(\mathbf{x})=\{m_1,...m_k,...m_K\}$ $(k\in K)$ *i.e.*, action and critic $V(\mathbf{x})$ *i.e.*, state value are calculated by Eqs. (12)-(14).

$$a_i=\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\sum\nolimits_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right) \tag{9}$$

$$\sum\nolimits_i=diag(\boldsymbol{\sigma}_i^2) \tag{10}$$

$$b_i=\frac{a_i}{\sum_{t=1}^{N_L}a_t} \tag{11}$$

$$m_k(\mathbf{x})=m_k^{\max}f_{sig}\left(\sum_{i=1}^{N_L}w_{ki}b_i+\beta n_k\right) \tag{12}$$

$$f_{sig}(z) = \frac{2}{1-\exp(-z)} - 1 \tag{13}$$

$$V(\mathbf{x}) = \sum_{i=1}^{N_L} v_i b_i \tag{14}$$

Here, i, j, k denote the subscripts of the basis functions of the agent, inputs and actor outputs, respectively. J, K also denote the dimensions of the inputs and actor outputs. In this chapter, the input of the statistical model is defined as becoming equal to that of the RL agent. In other words, the RL agent outputs the control bias to the input condition $\mathbf{x}$. The reward is calculated based on the results of control, *i.e.*, the outputs of the statistical model obtained after the control.

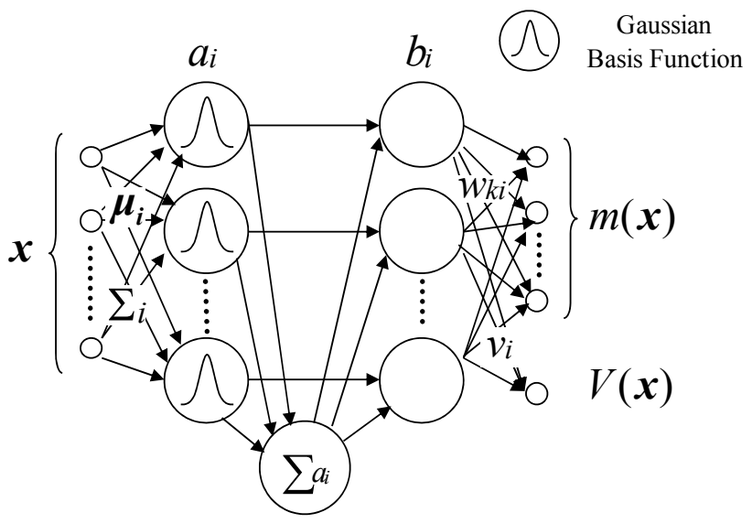


Fig. 3. Basic Structure of NGnet

Here, $\sum_i$ is the covariance matrix of the Gaussian basis function. $\boldsymbol{\mu}_i = \{\mu_{i1}, ...\mu_{ij}, ...\mu_{iJ}\}$, $\boldsymbol{\sigma}_i^2 = \{\sigma_{i1}^2, ...\sigma_{ij}^2, ...\sigma_{iJ}^2\}$ are the center and radii vectors, respectively. N_L is the basis function size. w_{ki}, v_i are the weight parameters of actor and critic, respectively.

The procedures to calculate the actor outputs m_k are as follows. First, the sum of the normalized activations b_i is added to a noise component to search for optimal actions. Next, they are converted to the region of [–1.1] by a sigmoid function. Finally, the actor outputs m_k are calculated by multiplying the maximum values of the actor outputs $m_k^{\max}$ and the converted value. Here, n_k is normalized noise whose average is 0 and variance is 1. β is a noise ratio.

2.3.2 Learning algorithm

Learning of NGnet is executed by the following procedures: updating the weight parameters w_{ki}, v_i, adding/deleting of the Gaussian basis functions, and tuning $\boldsymbol{\mu}_i$, $\boldsymbol{\sigma}_i^2$. TD

learning[17] is employed to update w_{ki}, v_i. The agent updates its input ($\mathbf{x} \rightarrow \mathbf{x}'$) by its actor outputs m_k, then the model outputs are calculated by the actor outputs. Eq. (15) calculates TD error δ by *reward* calculated by the model outputs and the state value $V(\mathbf{x}')$ calculated by the input $\mathbf{x}'$.

$$\delta = reward + \gamma V(\mathbf{x}') - V(\mathbf{x}) \tag{15}$$

Here, γ is a discount ratio for the future reward. The actor of NGnet learns its actions to improve $V(\mathbf{x})$, and the critic of NGnet also learns to estimate $V(\mathbf{x})$ appropriately. w_{ki}, v_i are updated by Eqs. (16) and (17) using δ.

$$w_{ki} = w_{ki} + \alpha_A b_i \delta n_k \tag{16}$$

$$v_i = v_i + \alpha_C b_i \delta \tag{17}$$

Here, α_A and α_C denote the learning rates of w_{ki} and v_i, respectively.
The other learning procedures execute adding/deleting the Gaussian basis functions and tuning of $\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2$ so that the NGnet can obtain enough resolutions to learn its state space. The proposed control system employs the following algorithm: the sizes of basis functions of the NGnet are initialized to 0, and new basis functions are added adaptively in its learning.

Basis Addition Algorithm

Step 1. If the current basis function size N_L satisfies $N_L < N_L^{\max}$, then the algorithm goes to **Step 2**. Otherwise, it terminates.

Step 2. The activations of the agent's current basis functions a_i are calculated for the input $\mathbf{x}$ during its learning.

Step 3. If there is no basis function i which meets $a_i \geq a_{\min}$, then the algorithm goes to **Step 4.** Otherwise, it terminates.

Step 4. If $\delta > \delta_{\min}$ is satisfied, the algorithm goes to **Step 5**. Otherwise, it terminates.

Step 5. A basis function whose center and radius is set to $\mathbf{x}$ and $\boldsymbol{\sigma}_i$ is added to NGnet, then the algorithm terminates.

Here, $N_L^{\max}, a_{\min}$ and $\delta_{\min}$ denote maximum basis function size, threshold value of activation and threshold value of TD error, respectively. This algorithm adds new basis functions in the regions of the state space which are not sufficiently covered with learned basis functions. In addition, the maximum basis function size $N_L^{\max}$ is set because it might be possible to add unnecessary basis functions by increasing variation of the TD error due to the proposed automatic reward adjustment method described later. Therefore, the agent can put only the necessary basis functions in its state space.

2.4 Learning flow of the proposed control system

The learning algorithm flow of the proposed control system consists of the following steps.

Learning Algorithm of the Proposed Control System

Step 1. Initialize learning parameters of the RBF network and RL.

Step 2. Adjust radii of the RBF network.
Step 3. Calculate weight parameters of the RBF network.
Step 4. Determine initial control variables.
Step 5. Change control variables by the RL agent.
Step 6. Calculate model outputs by the RBF network.
Step 7. Calculate reward.
Step 8. Calculate TD error.
Step 9. Update weight parameters of the RL agent.
Step 10. Add new basis functions of the RL agent.
Step 11. If the terminal condition of the episode is reached, go to **Step 12**. Otherwise, return to **Step 5**.
Step 12. Adjust the reward parameters.
Step 13. If the terminal condition of learning is reached, terminate the algorithm. Otherwise, return to **Step 4**.

In the above algorithm, an episode terminates after executing the processes between **Step 5** and **Step 10** for S times, and a trial of learning terminates after executing the processes between **Step 4** and **Step 12** for T times.

3. Adaptive radius adjustment method

3.1 Basic concepts

In the proposed control system, the outputs of the RBF network are calculated by the Gaussian basis functions according to the input space. To obtain high estimation accuracy, the radii should be adjusted so that the basis functions can cover the space sufficiently.

The proposed method focuses on the covering rate of the basis functions on the input space. It adjusts the radii based on the distances between a randomly generated input and the center of the basis functions selected to surround the input, where the learning data are located. As a result, the radii of basis functions whose distances to other data are short become small, and *vice versa*.

3.2 Algorithm of the proposed method

The algorithm of the proposed method consists of the following steps.

Algorithm of the Radius Adjustment Method

Step 1. Initialize the radii and adjusting parameters.
Step 2. Generate an input randomly.
Step 3. Select pairs of learning data by the *k*-SN (*k*-surrounded neighbor) method[24].
Step 4. Exclude the selected data from the data candidates for selection.
Step 5. If there are no data candidates, go to **Step 4.** Otherwise, return to **Step 3.**
Step 6. If there are no selected data, go to **Step 8**. Otherwise, go to **Step 7.**
Step 7. Update radii of the selected data
Step 8. If n reaches N, terminate the algorithm. Otherwise, increment n and return to **Step 2.**

In **Step 1**, the radii are initialized as a small value. In **Step 2**, an input condition $\mathbf{x}_n$ (n: suffix showing the number of iterations) is generated randomly. **In Step 3**, the pairs of learning data ($\mathbf{x}_m^1$, $\mathbf{x}_m^2$) (m: suffix showing the number of pairs) for which the radii are to be

adjusted are selected for the generated $\mathbf{x}_n$ using the *k*-SN method. The *k*-SN method is a data extraction method to satisfy the condition of interpolation. It selects the pair of data $(\mathbf{x}_m^1, \mathbf{x}_m^2)$ so that $\mathbf{x}_n$ is surrounded by them.

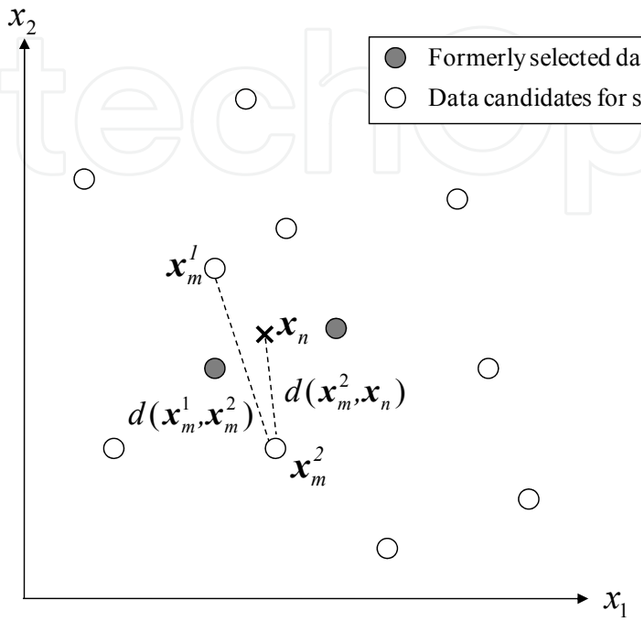


Fig. 4. Mechanism of k-SN method

Figure 4 shows the mechanism of the *k*-SN method in a 2-dimensional input space. The nearest datum $\mathbf{x}_m^1$ to $\mathbf{x}_n$ is selected from the data candidates available for selection, *i.e.*, learning data excluding the formerly selected data. Then the datum $\mathbf{x}_m^2$ paired with $\mathbf{x}_m^1$ is selected according to Eq. (18).

$$\mathbf{x}_m^2 = \arg\min_{z \in Z} d(\mathbf{x}_z, \mathbf{x}_n)$$
$$subject \quad to \quad d(\mathbf{x}_z, \mathbf{x}_n) < d(\mathbf{x}_m^1, \mathbf{x}_z) \tag{18}$$

Here, z denotes the suffix of the data candidates available for selection and $d(\mathbf{x}_z, \mathbf{x}_n)$ denotes the distance between $\mathbf{x}_z$ and $\mathbf{x}_n$. In **Step 4**, the selected data $(\mathbf{x}_m^1, \mathbf{x}_m^2)$ are excluded from the data candidates. If there is no $\mathbf{x}_z$ satisfying Eq. (18), only $\mathbf{x}_m^1$ is excluded. In this way, the radii of basis functions in an interpolative relation with inputs are adjusted, then the basis functions can cover the input space sufficiently. This selection continues until all the data candidates have been selected.

In **Step 7**, the radii (r_m^1, r_m^2) set at the selected data are adjusted by Eqs. (19) and (20).

$$r_m^1 = r_m^1 + \alpha_{rad} \tau^n \left(d(\mathbf{x}_m^1, \mathbf{x}_n) - r_m^1 \right) \tag{19}$$

$$r_m^2 = r_m^2 + \alpha_{rad}\tau^n \left(d(\mathbf{x}_m^2, \mathbf{x}_n) - r_m^2 \right) \tag{20}$$

Here, α_{rad} is an initial step size parameter of radius, and τ is a decay rate of the step size parameter ($0 < \tau < 1$). The second term in the right sides of both Eqs. (19) and (20) decays as iteration n increases, then the radii finally converge to certain values. These steps are iterated until n reaches N , then the radii are adjusted to certain values according to the distribution of learning data.

3.3 Simulations

In this section, some simulations are executed in order to evaluate the performances of the proposed radius adjustment method. The proposed method is compared with two conventional radius adjustment methods with respect to estimation accuracy and computational time using the test function data.

3.3.1 Simulation conditions

Simulations are executed in the following steps: a) determination of radii, b) calculation of weight parameters, and c) evaluation of estimation error. In step a), the proposed method, the Cross Validation (CV) method[11] and the radius equation method[20] are used to determine radii. The CV method adjusts radii with regression, and the radius equation method adjusts radii without regression. (See appendix). In step b), the weight parameters of the RBF network are calculated by Eq. (8). In step c), the estimation errors between the outputs of the RBF network and the test data are evaluated.

In the case of plant control, the shape of the response surface changes according to the plant properties, input dimensions and numbers of learning data. In order to simulate various response surfaces, the learning data are created for different test functions, input dimensions and numbers of data. The test functions $F_1(\mathbf{x})$ and $F_2(\mathbf{x})$ ($\mathbf{x} \in [-5,5]$) described as Eqs. (21) and (22) are used in the simulations. These functions are often used as benchmark problems of RBF networks[20].

$$F_1(\mathbf{x}) = \sum_{j=1}^{J} \left(x_j^4 - 16x_j^2 + 5x_j + 100 \right) \tag{21}$$

$$F_2(\mathbf{x}) = \sum_{j=1}^{J} \left(\sum_{k=1}^{j} x_k \right)^2 \tag{22}$$

Table 1 shows settings of learning data and test data of the RBF network in the simulations. Here, the numbers of learning data and test data are denoted as N_D and N_{Test} , respectively. In simulations, the output dimension P is fixed to 1, while the input dimension J is varied from 2 to 10. The parameters of α_{rad} , τ and N are set to 0.01, 0.999 and 3000, respectively. They are set appropriately based on prior experimental results. The parameters of $r^{\min}$, $r^{\max}$ and Δr used in the CV method are shown in Table 2. The common parameter, λ is set to 0.01. Each simulation is executed for 25 random sequences using a Linux machine (CPU clock: 2.8[GHz]).

Case	Function	Input Dimension J	Data Size	
			N_D	N_{Test}
1	$F_1(\boldsymbol{x})$	2	25	25
2			50	
3			100	
4		5	100	50
5			300	
6			500	
7		10	100	100
8			500	
9			1000	
10	$F_2(\boldsymbol{x})$	2	25	25
11			50	
12			100	
13		5	100	50
14			300	
15			500	
16		10	100	100
17			500	
18			1000	

Table 1. Specifications of learning data and test data in the simulation cases

Case	r^{min}	r^{max}	Δr
1,2,3,10,11,12	0.1	10	0.1
4,13	5	15	0.1
5,14	5	15	0.5
6,15	5	15	1
7,16	5	20	0.1
8,9,17,18	5	20	1

Table 2. Parameter conditions in CV method

3.3.2 Results and discussions

In order to evaluate estimation accuracy of the proposed method, root mean square error $RMSE_{cn}$ calculated by Eq. (23) is used.

$$RMSE_{cn} = \sqrt{\sum_{t=1}^{N_{Test}} \frac{\left(y_{cn}(\mathbf{x}_t) - F_{cn}(\mathbf{x}_t)\right)^2}{N_{Test}}} \tag{23}$$

Here, $y_{cn}(\mathbf{x}_t)$ and $F_{cn}(\mathbf{x}_t)$ are an output value of the RBF network and the test data for the input $\mathbf{x}_t$ (t : suffix of the test data) in case cn ($cn = \{1,2,\ldots 18\}$), respectively.

First, convergence performance of the proposed method is studied using the RMSE and adjusted radii parameters. Figs. 5 and 6 show the RMSE and several radii parameters for iteration N in case 5 of Table 1. Case 5 is the most suitable condition for real plants with respect to input dimensions and numbers of learning data. The other cases also show the results similar to those of case 5. From Fig. 5, it is confirmed that the RMSE decreases and converges into a certain value with iteration.

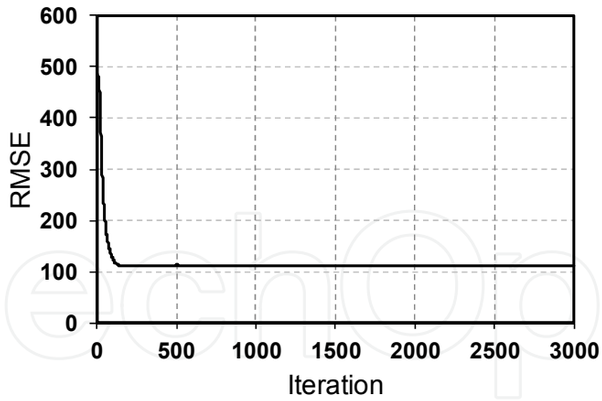


Fig. 5. RMSE curve obtained by the proposed method

Figure 6 shows the adjustment history of 10 typical radii selected from those of 300 Gaussian basis functions corresponding to the numbers of learning data in case 5. In this figure, the radii soon increase with iteration but converge into different values. The reason why the adjusted radii converge into different values is that the proposed method adjusts the radii based on the distribution of learning data. For the data whose distances to other data are short, the distances between the learning data and $\mathbf{x}_n$ become short. Consequently, the radii of the data in the region become shorter than those in the region whose distances are long. It is also confirmed by comparing Figs. 5 and 6 that the convergence of radii due to the decay of τ^n contributes to the convergence of RMSE.

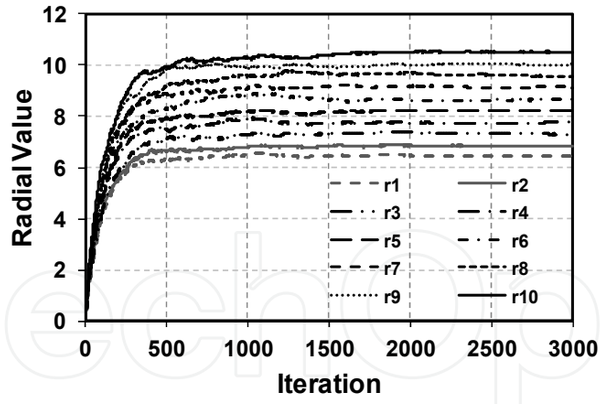


Fig. 6. Adjusting history of typical radial values by the proposed method

Next, Fig. 7 shows the radial values plotted for the crowding index ci of their basis functions calculated by Eqs. (24) and (25). The crowded index ci represents how the center coordinate $\mathbf{c}_i$ of the basis function i is covered with all the basis functions having uniform radii, thus this index of the data whose distances to other data are short usually becomes large.

$$h_t(\mathbf{c}_i) = \exp\left(-\frac{(\mathbf{c}_i - \mathbf{c}_t)^T (\mathbf{c}_i - \mathbf{c}_t)}{r_{ci}^2} \right) \tag{24}$$

$$ci(\mathbf{c}_i) = \frac{1}{N_D} \sum_{t=1}^{N_D} h_t(\mathbf{c}_i) \tag{25}$$

Here, $h_t(\mathbf{c}_i)$ is the Gaussian function value of the basis function whose center is $\mathbf{c}_t$ and r_{ci} is the radius to which a certain constant value is set. In this simulation, r_{ci} is set to 3.89 considering the range of input values.

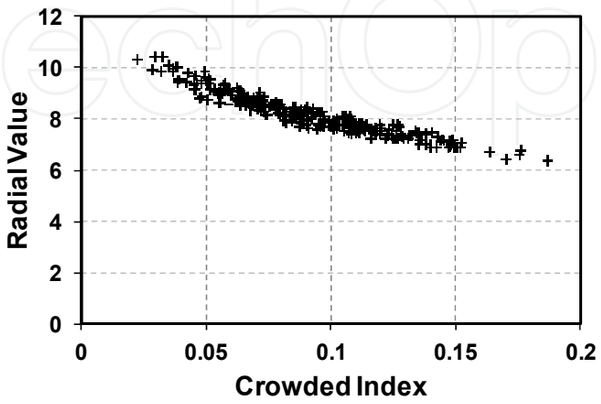


Fig. 7. Relation between the crowded index and radial values

In Fig. 7, the radial values with low crowded index are larger than those with high crowded index because the basis functions in the region where distances to data are long need to cover a wider input space. This result indicates that the proposed radius adjustment algorithm works properly.

Case	Proposed Method	CV Method	Radius Equation
1	76.0	83.5	96.5
2	71.4	70.9	99.7
3	29.5	36.1	63.3
4	138.3	130.3	186.5
5	116.6	115.4	170.1
6	107.2	113.3	153.1
7	201.0	234.1	272.7
8	174.4	166.8	230.7
9	164.0	158.4	239.3
10	7.9	6.1	14.2
11	2.7	2.7	10.5
12	1.5	2.3	9.3
13	39.8	35.3	79.1
14	16.4	12.9	58.5
15	8.9	10.9	48.4
16	268.1	292.2	294.7
17	82.0	58.8	173.5
18	63.9	38.2	175.6
Ave.	87.3	87.1	132.0

Table 3. Comparisons of the RMSEs obtained by the proposed and conventional methods

Table 3 compares the RMSEs of the proposed method and conventional methods. The case values in the table are the averages of 25 simulation results. The RMSEs of the proposed method are smaller than those for the radius equation in each case. The radius equation is usually applied to learning data having a uniform crowded index[20]. Therefore, it is difficult to apply it to plant control where the learning data usually have deviations of crowded index like Fig. 7. The proposed method can adjust the radii considering the distribution of the learning data, thus the RMSEs are an average of 33.9[%] better compared to those from the radius equation. The proposed method also has the same performances as the CV method.

Table 4 compares computational times of the proposed and conventional methods. These case results are also the averages of 25 simulation results. The computational times of the radius equation are enormously short because it spends time only in the calculation of Eq. (34) to adjust the radii. Regarding the CV method, the computational times increase exponentially with the number of data because error evaluations are needed for all learning data. There are some cases where the computational times are well beyond the limitation of practical use (20 minutes). Therefore, it is difficult to apply the CV method to plant control. On the other hand, the computational times of the proposed method in every case are within 20 minutes. These computational times are practical for plant control and it is confirmed that the proposed method is the most suitable for plant control.

These simulation results show that the proposed plant control system can construct a flexible statistical model having high estimation accuracy for various operational conditions of thermal power plants within a practical computational time. It is expected to improve effectiveness in reducing NOx and CO by learning with such a statistical model.

Case	Proposed Method	CV Method	Radius Equation
1	2.8E-02	6.5E-01	7.6E-06
2	9.9E-02	9.2E+00	2.8E-05
3	3.7E-01	1.5E+02	1.1E-04
4	4.6E-01	1.4E+02	1.4E-04
5	3.9E+00	2.6E+03	1.3E-03
6	1.1E+01	1.7E+04	3.6E-03
7	6.6E-01	2.2E+02	2.8E-04
8	1.6E+01	2.3E+04	6.9E-03
9	6.4E+02	6.5E+05	3.1E-02
10	2.7E-02	6.5E-01	7.6E-06
11	9.8E-02	9.2E+00	2.7E-05
12	3.7E-01	1.5E+02	1.1E-04
13	4.6E-01	1.4E+02	1.4E-04
14	3.9E+00	2.6E+03	1.3E-03
15	1.1E+01	1.6E+04	3.6E-03
16	6.6E-01	2.2E+02	2.8E-04
17	1.6E+01	2.3E+04	6.9E-03
18	6.4E+02	6.5E+05	3.1E-02

Table 4. Comparisons of the computational times [s] for the proposed and conventional methods

4. Automatic reward adjustment method

4.1 Basic concepts

When the RL is applied to the thermal power plant control, it is necessary to design the reward so that it can be given to the agent instantly in order to adapt to the plant properties which change from hour to hour. So far, studies with respect to designing reward of the RL have reported[25,26] that high flexibility could be realized by switching or adjusting the reward in accordance with change of the agent's objectives and situations. However, it would be difficult to apply this to thermal power plant control which needs instant reward designing for changes of plant properties because the reward design and its switching or adjusting depend on *a priori* knowledge.

The proposed control system defines a reward function which does not depend on the learning object and proposes an automatic reward adjustment method which adjusts the parameters of the reward function adaptively based on the plant property information obtained in the learning. It is possible to use the same reward function for different operating conditions and control objectives in this method, and the reward function is adjusted in accordance with learning progress. Therefore, it is expected possible to construct a flexible plant control system without manual reward design.

4.2 Definition of reward

The statistical model in the proposed control system has a unique characteristic due to specifications of applied plants, kinds of environmental effects and operating conditions. In case such a model is used for learning, the reward function should be generalized because it is difficult to design unique reward functions for various plant properties in real time. Thus the authors have defined the reward function as Eq. (26).

$$reward = \begin{cases} reward_{\max} \exp\left(\dfrac{\rho - f}{\phi}\right) & (f \geq \rho) \\ reward_{\max} & (f < \rho) \end{cases} \tag{26}$$

Here, $reward_{\max}$ and f are maximum reward value and sum of weighted model outputs calculated by Eq. (27), respectively. ϕ and ρ are the parameters to determine shapes of the reward function.

$$f = \sum_{p=1}^{P} C_p y_p \tag{27}$$

Here, C_p are the weight of the model output y_p, and p is a suffix for model output. In Eq. (26), the conditions $\phi > 0$, $\rho \geq 0$ are satisfied. If ϕ and ρ become larger, a larger reward is gotten for f. In addition, it is possible for f to weight y_p by C_p in accordance with control goals. Fig. 8 shows the shape of the reward function where $reward_{\max} = 1$, $\phi = 10$, $\rho = 20$ are set in Eq. (26).

The reward function defined as Eq. (26) can be applied for various kinds of statistical models where the operating conditions and the control goals are different because it is possible to define the reward only by ϕ, ρ and C_p. C_p is set in accordance with the control goals, and ϕ, ρ are adjusted automatically by the proposed automatic reward adjustment method.

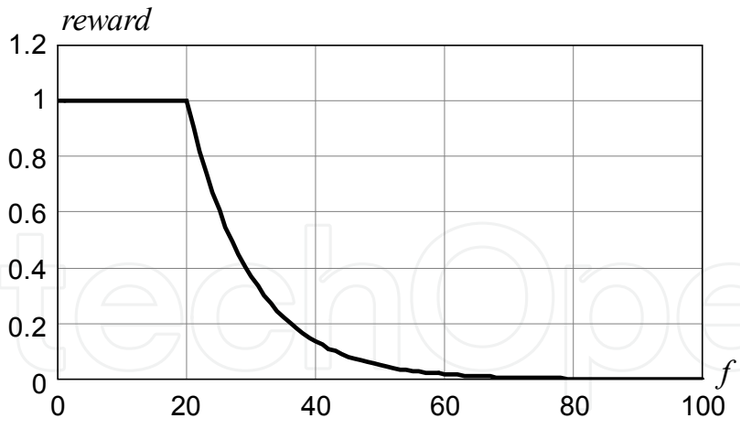


Fig. 8. Schematic of reward function

4.3 Algorithm of the proposed reward adjustment method

The proposed reward adjustment method adjusts the reward parameters ϕ, ρ using the model outputs which are obtained during the learning so that the agent can get the proper reward for (1) characteristics of the learning object and (2) progress of learning. Here, (1) means that this method can adjust the reward properly for the statistical models whose optimal control conditions and NOx/CO properties are different by adjusting ϕ, ρ. (2) means that this method makes it easier for the agent to get the reward and accelerate learning at the early stage, while also making the conditions to get the reward stricter and improving the agent's learning accuracy.

The reward parameters are updated based on the sum of weighted model outputs f obtained in each episode and the best f value obtained during the past episodes. Hereafter, the sum of weighted model outputs and the reward parameters at episode t are denoted as f_t, ϕ_t and ρ_t, respectively.

The algorithm of the proposed method is as follows. First, f_t is calculated by Eq. (28), then its moving average $\overline{f_t}$ is calculated.

$$\overline{f_t} = \varepsilon f_t + (1-\varepsilon)\overline{f_{t-1}} \tag{28}$$

Here, ε is a smoothing parameter of the moving average. The parameter ϕ_t is updated by Eqs. (29) and (30) where $\overline{f_t} > \rho_t$ is satisfied.

$$\phi_{t+1} = \phi_t + \alpha_\phi(\phi'_t - \phi_t) \tag{29}$$

$$\phi'_t = \frac{\rho_t - \overline{f_t}}{\ln(\theta_t / reward_{\max})} \tag{30}$$

Here, ϕ'_t is an updating index of ϕ_t, θ_t is a threshold parameter to determine the updating direction (positive/negative), and α_φ is a step size parameter of ϕ_t. As shown in Fig. 9, ϕ'_t corresponds to the ϕ when the reward value for $\overline{f_t}$ becomes θ_t. The updating direction of ϕ_t becomes positive where θ'_t calculated by Eq. (31) is smaller than θ_t, and *vice versa*.

$$\theta_t' = reward_{\max} \exp\left(\frac{\rho_t - \overline{f_t}}{\phi_t} \right) \tag{31}$$

θ_t is updated by Eq. (32) so that it becomes closer to θ_t'.

$$\theta_{t+1} = \theta_t + \alpha_\theta (\theta_t' - \theta_t) \tag{32}$$

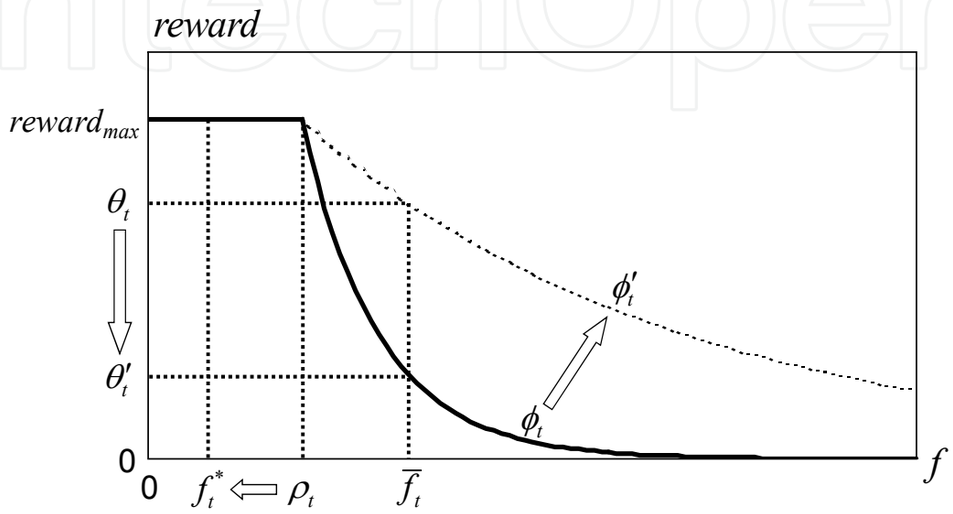


Fig. 9. Mechanism of the proposed method

Here, α_θ is a step size parameter of θ_t. θ_t is initialized to small value. As a result of updating θ_t by Eq. (32), finally ϕ_t' becomes equal to ϕ_t. This means that the reward is given to the agent appropriately for current $\overline{f_t}$. The value of θ_t depends on the learning object and progress, hence it is preferable to acquire empirically in the learning process. That is because θ_t', the reward value for $\overline{f_t}$ is defined according to the updating index of θ_t.
The parameter ρ_t is updated to approach the f_t^* by Eq. (33) which is the best value of f during past learning.

$$\rho_{t+1} = \rho_t + \alpha_\rho (f_t^* - \rho_t) \tag{33}$$

Here, α_ρ is a step size parameter of ρ_t.

The above algorithm is summarized as the following steps.

Reward Automatic Adjustment Algorithm

Step 1. Calculate $\overline{f_t}$ by Eq. (28).

Step 2. If $\overline{f_t} > \rho_t$ is satisfied, go to **Step 3**. Otherwise, go to **Step 5**.

Step 3. Update ϕ_t by Eqs. (29) and (30).

Step 4. Update θ_t by Eqs. (31) and (32).

Step 5. Update ρ_t by Eq. (33) and terminate the algorithm.

4.4 Simulations

In this section, simulations are described to evaluate the performances of the proposed control system with the automatic reward adjustment method when it is applied to virtual plant models configured on the basis of experimental data. The simulations incorporate changes of the plant operations several times and the data for the RBF network. The evaluations focus on the flexibility in control of the proposed reward adjustment method for the change of the operational conditions. In addition, the robustness in control for the statistical model including noise by tuning the weight decay parameter of RBF network is also studied.

4.4.1 Simulation conditions

Figure 10 shows the basic structure of the simulation. The objective of the simulation is to reduce NOx and CO emissions from a virtual coal-fired boiler model (statistical model) constructed with three numerical calculation DBs. The RL agent learns how to control three operational parameters with respect to air mass flow supplied to the boiler. Therefore, input and output dimensions (J, P) of the control system are 3 and 2, respectively. The input values are normalized into the range of $[0,1]$. The three numerical calculation DBs have different operational conditions, and each DB has 63 data whose input-output conditions are different. These data include some noise similar to the actual plant data.

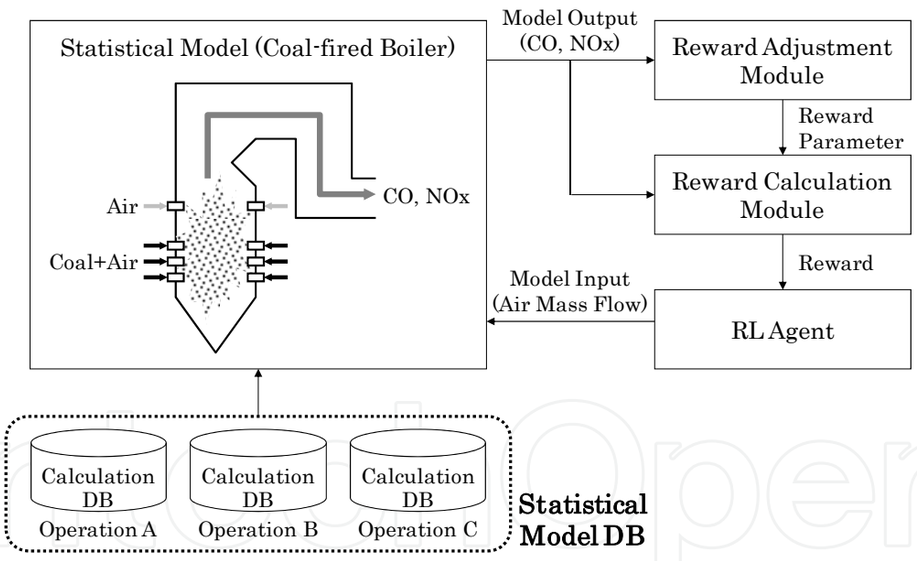


Fig. 10. Basic structure of thermal power plant control simulation

In this simulation, the robustness and flexibility of the proposed control system are verified by implementing the RL agent so that it learns and controls the statistical model which changes in time series. Two kinds of boiler operational simulations are executed according to Table 5. Each simulation case is done for six hours (0:00-6:00) of operation, and it is considered that the statistical model is changed at 0:00, 2:00 and 4:00. One of the simulations considers three kinds of operational conditions (A, B, C) where coal types and power outputs are different, and the other considers three kinds of control goals defined as Eq. (27), where the weight coefficients C_1, C_2 of CO and NOx, respectively in that equation are different.

The simulations are executed by two reward settings: the variable reward for the proposed reward adjustment method (proposed method) and the fixed reward (conventional method). Both reward settings are done under two conditions where the weight decay λ for the RBF network is set to 0, 0.01 to evaluate the robustness of control by λ settings. The RL agent learns at the times when operational conditions or control goals (0:00, 2:00 and 4:00) are changed, and the control interval is 10 minutes. Hence it is possible to control the boiler 11 times in each period.

Parameter conditions of learning are shown in Table 6. These conditions are set using prior experimental results. The parameter conditions of reward are shown in Table 7. The parameters ($\varepsilon, \alpha_\phi, \alpha_\theta, \alpha_\rho$) of the proposed method are also set properly using prior experiments. In the conventional method, the values of ϕ, ρ are fixed to their initial values which are optimal for the first operational condition in Table 5 because their step size parameters (α_ϕ, α_ρ) are set to 0.

Objective	Change of Operational Conditions			Change of Goals		
Time	Ope. Cond.	C_1	C_2	Ope. Cond.	C_1	C_2
0:00 - 2:00	A	0.1	0.9	A	0.1	0.9
2:00 - 4:00	B	0.1	0.9	A	0.9	0.1
4:00 - 6:00	C	0.1	0.9	A	0.001	0.999

Table 5. Time table of plant operation simulation

Parameter		Condition
Radius of Gaussian basis	σ	0.2
Max. output of NGnet	$m_k^{\max}$	0.2
Noise ratio	β	0.2
Discount rate	γ	0.9
Learning rate for actor	α_A	0.1
Learning rate for critic	α_C	0.02
Max. basis num of agent	$N_I^{\max}$	100
Min. a_i for basis addition	$a_{\min}$	0.368
Min. δ for basis addition	$\delta_{\min}$	0.01
Max. iteration in 1 episode	S	30
Max. episode	T	10000

Table 6. Parameter conditions of learning

Parameter		Prop. Method	Conv. Method
Max. reward	$reward_{\max}$	1	1
Smoothing parameter	ε	0.1	0.1
Step size parameter of ϕ	α_ϕ	0.05	0
Step size parameter of ρ	α_ρ	0.05	0
Step size parameter of θ	α_θ	0.05	0
Initial value of ϕ		0.001	3
Initial value of ρ		0.001	0
Initial value of θ		0	186

Table 7. Reward conditions of each method

4.4.2 Results and discussion

Figure 11 shows the time series of normalized f as a result of controls by the two methods, where the initial value at 0:00 is determined as the base. There are four graphs in Fig. 11 with combinations of the two objectives of simulations and λ settings. The optimal f value in each period is shown as well. The computational time of learning in each case was 23[s].

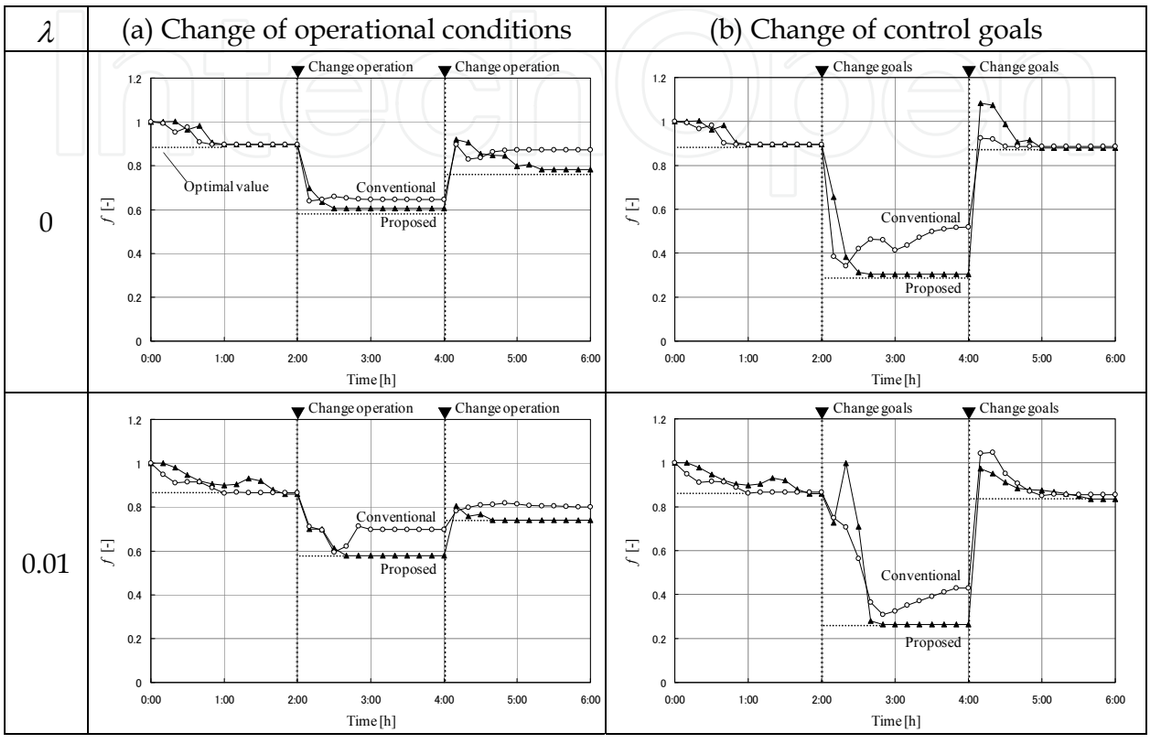
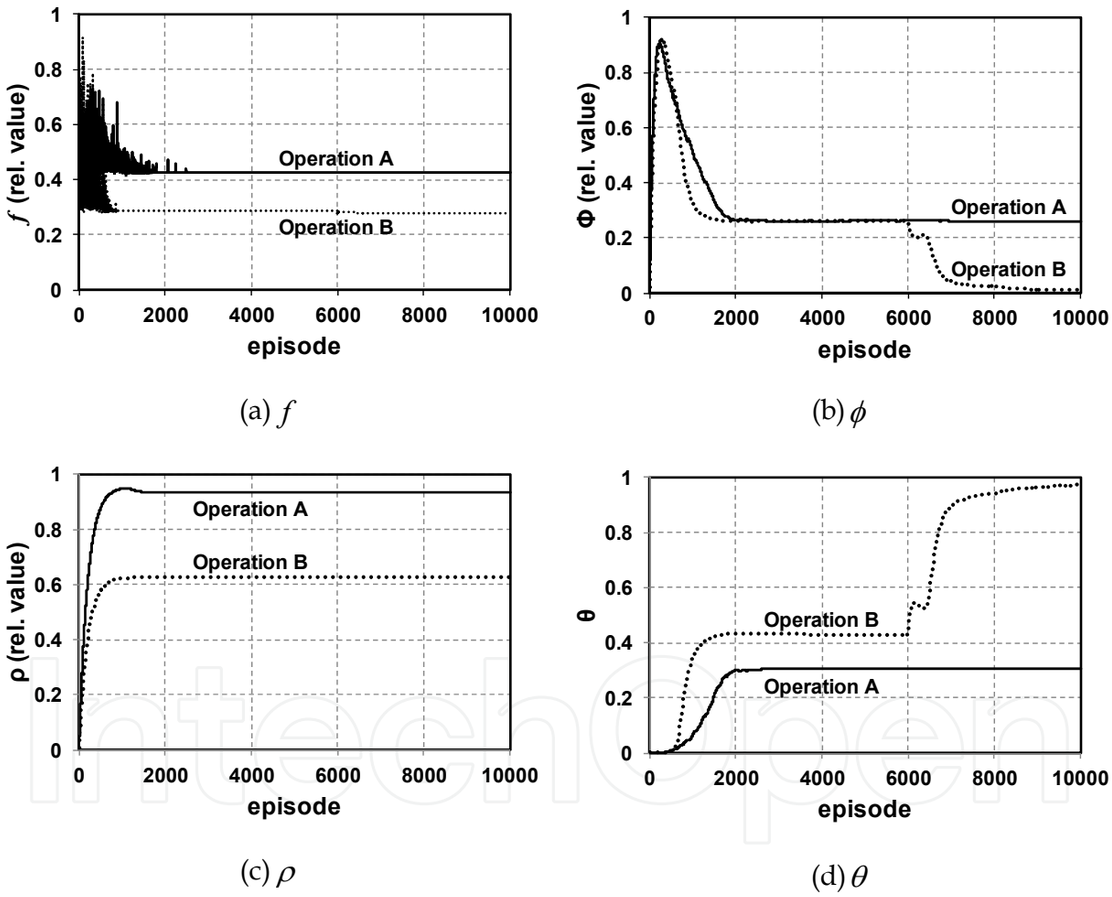


Fig. 11. Time series of normalized f in the boiler operation simulations

To begin with, time series of the normalized f values by the proposed method and conventional method in the case of λ=0.01 are discussed. The initial f values at 0:00 of these methods have offsets with the optimal values, but they are decreased for control and finally converged near the optimal values. This is because the reward functions used in each method are appropriate to learn the optimal control logic. The RL agent relearns its control logic when the statistical model and its optimal f values are changed at 2:00 by the change of operational conditions or control goals. However, the f values of the conventional method after 11 control times still have offsets from the optimal values, while the proposed method can obtain the optimal values after 11 times. The initial reward setting of the conventional method would be inappropriate for the next operational condition. Similar results of control are obtained for the same reason after changing the statistical model at 4:00. As discussed above, the plant control system by the conventional method has a possibility to deteriorate the control performances in thermal power plants for which operational conditions and control goals are changed frequently. Therefore, the proposed reward adjustment method is effective for the plant control, which can adjust the reward function flexibly for such changes.

Next, the robustness of the proposed control system by weight decay (λ) tuning is discussed. In Fig. 11, every f value of the proposed method can reach nearly the optimal value when λ is 0.01, whereas f converges into the values larger than the optimal values when λ is 0 for 2:00-6:00 in (a) and 2:00~4:00 in (b). The RBF network cannot learn with considered the influences of noise included in the learning data when λ is 0[16]. The response surface is created to fit the noised data closely and many local minimum values are generated in it compared with the response surface of $\lambda = 0.01$. This is because the learned control logic is converged each local minimum. The above results show that the RBF network can avoid overfitting by tuning λ properly and the proposed control system can control thermal power plants robustly.



(a) f (b) ϕ

(c) ρ (d) θ

Fig. 12. Learning processes of f and reward parameters (ϕ, ρ, θ) of the proposed method

Finally, the learning processes of f and reward parameters of the proposed method are studied. Fig. 12 shows the f, ϕ, ρ, θ values for episodes in learning at the operational changes at 0:00 and 2:00 when λ is 0.01. In the early stage of learning (episodes 1-500), the ϕ parameter in each case increases nearby 0.9 because the f value does not decrease due to

insufficient learning of the RL agent. In the next 1000 episodes, ϕ increases and θ decreases simultaneously as the learning progresses. This behavior can be explained by the Eqs. (29)-(32) which are the updating algorithms of ϕ, θ. On the other hand, ρ value in each case converges to certain values by the 2000th episode. This indicates that the optimal f values are found in the learning process. Then the parameters of each case remain stable during the middle stage of learning (episode 2000-6000), but ϕ, θ change suddenly at the 6000th episode only in the case of operation B. This is because the RL agent can learn the control logic to get a better f value, then ϕ, θ are adjusted flexibly in accordance with the change of f used in Eqs. (29) and (30). As a result, these parameters converge into different values.
These adjustment results of reward parameters for different statistical models can be discussed as follows. By analysis of the characteristics of these statistical models, it seems that the gradient of f in operation A is larger than that of operation B because operation A has a larger difference between the maximum and minimum value of f than operation B. When the gradient of f is larger, f will vary significantly for each control thus it is necessary to set ϕ larger so that the agent can get the reward easily. On the other hand, it is useless to set ϕ larger in the statistical model in operation B for which the gradient of f is small. As for the results of adjustment of ϕ, ρ, θ in Fig. 12, the reward function of operation A certainly becomes easier to give the reward due to the larger ϕ than for operation B. Therefore, the above results show that the proposed method can obtain the appropriate reward function flexibly in accordance with the properties of the statistical models.

5. Conclusions

This chapter presented a plant control system to reduce NOx and CO emissions exhausted by thermal power plants. The proposed control system generates optimal control signals by that the RL agent which learns optimal control logic using the statistical model to estimate the NOx and CO properties. The proposed control system requires flexibility for the change of plant operation conditions and robustness for noise of the measured data. In addition, the statistical model should be able to be tuned by the measured data within a practical computational time. To overcome these problems the authors proposed two novel methods, the adaptive radius adjustment method of the RBF network and the automatic reward adjustment method.
The simulations clarified the proposed methods provided high estimation accuracy of the statistical model within practical computational time, flexible control by RL for various changes of plant properties and robustness for the plant data with noise. These advantages led to the conclusion that the proposed plant control system would be effective for reducing environmental effects.

6. Appendix A. Conventional radius adjustment method

A.1 Cross Validation (CV) method

The cross validation (CV) method is one of the conventional radius adjustment methods for the RBF network with regression and it adjusts radii by error evaluations. In this method, a datum is excluded from the learning data and the estimation error at the excluded datum is

evaluated. Iterations are repeated until all data are selected as excluded data to calculate RMSE. After the calculations of RMSE for several radius conditions, the best condition is determined as the radius to use. The algorithm is shown as follows.

Algorithm of Cross Validation Method

Step 1. Initialize the radius is initialized to $r^{\min}$.
Step 2. Select an excluded datum.
Step 3. Learn weight parameters of RBF network using all data except the excluded datum.
Step 4. Calculate the output of the RBF network at the point of the excluded datum.
Step 5. Calculate the error between the output and the excluded datum.
Step 6. Go to **Step 7** if all data have been selected. Otherwise, return to **Step 2**.
Step 7. Calculate RMSE by the estimation errors.
Step 8. Increment the radius by Δr.
Step 9. Select the radius with the best RMSE if the radius is over $r^{\max}$ and terminate the
Step 10. algorithm. Otherwise, return to **Step 2**.

A.2 Radius equation

This method is one of the non-regression methods and it adjusts the radius r by Eq. (34).

$$r = \frac{d_{\max}}{\sqrt{J}\left(\sqrt[J]{N_D} - 1\right)} \tag{34}$$

Here, $d_{\max}$ is the maximum distance among the learning data.

7. References

[1] U.S. Environmental Protection Agency, Available from http://www.epa.gov/air/ oaq_caa.html/

[2] Ochi, K., Kiyama, K., Yoshizako, H., Okazaki, H. & Taniguchi, M. (2009), Latest Low-NOx Combustion Technology for Pulverized-coal-fired Boilers, *Hitachi Review*, Vol. 58, No. 5, pp. 187-193.

[3] Jorgensen, K. L., Dudek, S. A. & Hopkins, M. W. (2008), Use of Combustion Modeling in the Design and Development of Coal-Fired Furnaces and Boilers, *Proceedings of ASME International Mechanical Engineering Congress and Exposition,* Boston.

[4] EPRI (2005), *Power Plant Optimization Industry Experience*, 2005 Update. EPRI, Palo Alto.

[5] Rangaswamy, T. R.; Shanmugam J. & Mohammed K. P. (2005), Adaptive Fuzzy Tuned PID Controller for Combustion of Utility Boiler, *Control and Intelligent Systems*, Vol. 33, No. 1, pp. 63-71.

[6] Booth, R. C. & Roland W. B. (1998), Neural Network-Based Combustion Optimization Reduces NOx Emissions While Improving Performance, *Proceedings of Dynamic Modeling Control Applications for Industry Workshop*, pp.1-6.

[7] Radl B. J. (1999), Neural networks improve performance of coal-fired boilers, CADDET Energy Efficiency Newsletter, No.1, pp.4-6.

[8] Winn H. R. & Bolos H. R. (2008), Optimizing the Boiler Combustion Process in Tampa Electric Coal Fired Power Plants Utilizing Fuzzy Neural Model Technology, *Proceedings of Power-Gen International 2008*, Orlando, FL.

[9] Vesel R. (2008), The Million Dollar Annual Payback: Realtime Combustion Optimization with Advanced Multi-Variable Control at PPL Colstrip, *Proceedings of Power-Gen International 2008*, Orlando, FL.

[10] Airikka P. & Nieminen V. (2010), Optimized Combustion through Collaboration of Boiler and Automation Suppliers, *Proceedings of Power-Gen International 2008*, Amsterdam.

[11] Wasserman P. D. (1993), *Advanced Methods in Neural Computing*, Van Nostrand Reinhold.

[12] Rumelhart D. E.; Hinton G. E. & Williams R. J. (1986), Learning Representations of Back-propagation Errors, *Nature*, vol. 323, pp. 533-536.

[13] Camacho, E. F. & Bordons, C. (1999), *Model Predictive Control*. Springer.

[14] Jamshidi, M., Titli, A., Zadeh, L. & Boverie, S. (1997), *Applications of Fuzzy Logic*. Prentice Hall.

[15] Yamamoto K.; Fukuchi T.; Chaki M.; Shimogori Y. & Matsuda J. (2000), Development of Computer Program for Combustion Analysis in Pulverized Coal-fired Boilers, *Hitachi Review*, Vol. 49, No. 2, pp. 76-80.

[16] Orr M. J. L.. Introduction to Radial Basis Function Networks. Available from http://anc.ed.ac.uk/mjo/rbf.html

[17] Maruyama M. (1992), Learning Networks Using Radial Basis Function - New Approach for the Neural Computing. *Trans. of ISCIE*, Vol. 36, No. 5, pp. 322—329. (in Japanese)

[18] Sutton R. S. & Barto A. G. (1998), *Reinforcement Learning-An Introduction*, MIT Press.

[19] Bishop, C., M. (2006). *Pattern Recognition And Machine Learning*. Springer-Verlag.

[20] Kitayama S.; Yasuda K. & Yamazaki K. (2008), The Integrative Optimization by RBF Network and Particle Swarm Optimization, *IEEJ Trans. on EIS*, Vol. 128, No. 4, pp. 636-645. (in Japanese)

[21] Eguchi, T.; Sekiai T.; Yamada, A.; Shimizu S. & Fukai M. (2009), A Plant Control Technology Using Reinforcement Learning Method with Automatic Reward Adjustment, *IEEJ Trans. on EIS*, Vol. 129, No. 7, pp. 1253-1263. (in Japanese)

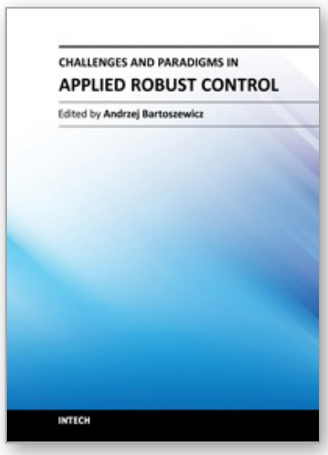
[22] Eguchi, T.; Sekiai T.; Yamada, A.; Shimizu S. & Fukai M. (2009), An Adaptive Radius Adjusting Method for RBF Networks Considering Data Densities and Its Application to Plant Control Technology, *Proceedings of ICCAS-SICE2009*, pp.4188-4194, Fukuoka, Japan, August 18-21.

[23] Moody J. & Darken C. J. (1989), Fast learning in networks of locally-tuned processing units, *Neural Computation*, Vol.1 , pp. 281-294.

[24] Zhang J.; Yim Y. & Yang J. (1997), Intelligent Selection of Instances for Prediction Function in Lazy Learning Algorithms, *Artificial Intelligence Review*, Vol. 11, pp. 175-191.

[25] Ng. A.; Harada D. & Russell S. (1999), Policy invariance under reward transformations: Theory and application to reward shaping, *Proceedings of 16th International Conference on Machine Learning*, pp.278-287.

[26] Li J. & Chan L. (2006), Reward Adjustment Reinforcement Learning for Risk-averse Asset Allocation, *Proceedings of International Joint Conference on Neural Networks 2006 (IJCNN06)*, pp.534-541.



Challenges and Paradigms in Applied Robust Control

Edited by Prof. Andrzej Bartoszewicz

ISBN 978-953-307-338-5
Hard cover, 460 pages
Publisher InTech
Published online 16, November, 2011
Published in print edition November, 2011

The main objective of this book is to present important challenges and paradigms in the field of applied robust control design and implementation. Book contains a broad range of well worked out, recent application studies which include but are not limited to H-infinity, sliding mode, robust PID and fault tolerant based control systems. The contributions enrich the current state of the art, and encourage new applications of robust control techniques in various engineering and non-engineering systems.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Toru Eguchi, Takaaki Sekiai, Naohiro Kusumi, Akihiro Yamada, Satoru Shimizu and Masayuki Fukai (2011). A Robust and Flexible Control System to Reduce Environmental Effects of Thermal Power Plants, Challenges and Paradigms in Applied Robust Control, Prof. Andrzej Bartoszewicz (Ed.), ISBN: 978-953-307-338-5, InTech, Available from: http://www.intechopen.com/books/challenges-and-paradigms-in-applied-robust-control/a-robust-and-flexible-control-system-to-reduce-environmental-effects-of-thermal-power-plants

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen