Control Using Neural Networks and Adaptive Control for Cooling Rate in GMA Welding

by

Isamu Okamura

Kogaku-shi (B.E.), University of Tokyo (1982)

Submitted to the Department of
Mechanical Engineering in Partial Fulfillment of
the Requirements for the Degree of

Master of Science
in Mechanical Engineering
at the

Massachusetts Institute of Technology

May 1994

Signature of Author.....
Department of Mechanical Engineering
May 6, 1994

Certified by .....
David E. Hardt
Professor, Department of Mechanical Engineering
Thesis Supervisor

Accepted by .....
Ain A. Sonin
Chairman, Departmental Committee on Graduate Studies

Control Using Neural Networks and Adaptive Control for Cooling Rate in GMA Welding

by

Isamu Okamura

Submitted to the Department of Mechanical Engineering
on May, 1994 in partial fulfillment of the requirements for
the Degree of Master of Science in Mechanical Engineering

Abstract

The dynamic behavior of cooling rate during gas metal arc welding (GMAW) has been studied experimentally using an infrared pyrometer which scans the weldment using an oscillating mirror. The purpose of this study is to learn how the input variables of wire feed rate and travel speed affect the cooling rate. The experiments show that the wire feed rate has little effect on the steady-state cooling rate, although a distinct change of the temperature distribution is observed. By contrast, the travel speed has a strong effect on the cooling rate.

A comparison between analytical models and the actual measurement is made. The analytical models are based on Green's function for the partial differential equations of the heat transfer. They do not succeed in explaining the small effect of the wire feed rate on the cooling rate. On the other hand, there is good correspondence between the analytical model and the actual measurement for the dynamic behavior of the cooling rate. The analytical model, together with the experiment results, suggests the cooling rate should be controlled by the travel speed. The control should also accommodate the nonlinearity and the varying time-delay that are observed.

Two types of control schemes, linear adaptive control and nonlinear control using neural networks are proposed to achieve high performance. The adaptive control adjusts the parameters of its linear model to the perturbation caused by the nonlinearity. The control using neural networks regulates the cooling rate by directly employing a nonlinear model. Thus, both schemes can accommodate the nonlinear nature of the process.

A one-step-ahead adaptive control method is used as the adaptive controller. Some practical considerations, such as a weighing factor, an integrator and a constrained parameter estimation are discussed. Two types of model structures are proposed to cope with the varying time-delay. A recursive least-square algorithm is used as the on-line estimator. The estimation by this algorithm is fast enough to accommodate the nonlinearity of the process. For the controller using neural networks, first the characteristics of neural networks are investigated. The investigation suggests control based on Smith's principle because of the process delays. The design methods, such as the parallel identification based on a conjugate gradient algorithm and the controller design based on a back-propagation method with a distal system, are proposed. Some limitations that come from the complexity of neural computation and nonlinear identification are found.

The performances of both control schemes are shown by simulation using the analytical model. Both control schemes bring about quite satisfactory results, showing better performance than PI control.

Thesis Supervisor: Dr. David E. Hardt
Title: Professor of Mechanical Engineering

Acknowledgments

I wish to express my deepest gratitude to Professor David E. Hardt, my thesis advisor, for giving me the opportunity to work on this project and learn from his expertise in process control. His guidance and suggestions were invaluable throughout my research. I also thank professor Hardt for his patience and time in reading this thesis.

I would like to thank other members of the faculty at MIT for giving me a chance to understand and learn theories associated with control. In particular, I would like to thank Professor Haruhiko Asada for his teaching and advise on the control using neural networks.

I also wish to thank Professor Toshiyuki Kitamori and Professor Shuji Yoshizawa at the University of Tokyo for their encouragement and advice for my study at MIT.

I would like to express my appreciation to my colleagues in our laboratory, Andrew Parris, Benny Budiman, Christopher Ratliff, Dan Walczyk, Robert Bakkestuen, and Upi Ummethala for their friendship and technical advice.

My special appreciation goes to my friends, Mr. David Mazzola, and Dr. Jang Boem Seong who offered me invaluable support of my life in the United States. Their heartwarming hospitality and encouragement kept me going through my career at MIT.

I am very thankful to my parents for their love, support, and encouragement through out my life. I would like to express my gratitude to my wife, Keiko and my children, Shun and Reina. Their assistance and love always have cheered me up, and led me to the goal of completion. I would like to dedicate this thesis to them.

Finally, I would like to gratefully acknowledge Kawasaki Steel Corporation for giving me the opportunity to study at MIT. Special thanks go to Mr. Kazushi Baba, and Mr. Tadaaki Iwamura for their encouragement and support.

This project was supported by the U.S. Department of Energy under contract number DE-FG02-85ER13331.

Table of contents

List of figures

List of Tables

Chapter 1

Introduction

Joining is an important and necessary aspect of manufacturing operations. It is achieved by various methods. Among them the arc welding is thought of as the most fundamental method for joining metals. The arc welding process involves partial melting and fusion of the joint between two workpieces, which are achieved by the thermal energy established between an electrode and materials. Therefore, the modification of both the geometry and properties of materials always takes place during welding. Weld quality takes metallurgical and mechanical features such as loading capacity, fracture toughness, oxidation and corrosion resistance, and geometric tolerance of the joint. The final objective of the welding control is, therefore, to establish a control system to get satisfactory results for these weld quality factors.

This thesis deals with the control for gas metal arc welding (GMAW). The control should have a feed-back form based on an in-process measurement. However, the above-mentioned weld quality factors are quite hard to measure. Thus, an indirect approach for controlling the weld quality is used. The key outputs for GMAW, which are associated with the weld quality, are three geometry features (bead width, height, and depth), and two thermal features (heat affected zone width and centerline cooling rate). A multivariable scheme for the control of these features has been presented by Hardt (1990). The typical inputs of this control are the travel speed of the torch and the wire feed rate as shown in Fig. 1.1.1. It is quite important to know and understand how such inputs affect these outputs.

Our research is motivated by the desire to control the mass transfer rate (mass fed into the workpiece per a unit length) and the centerline cooling rate. A control structure using fuzzy control and neural networks has been presented by Einerson et al for this problem (1992). Since the dynamic behavior of the system has not been discussed in this paper, we have a fundamental question as to the decoupleness of the mass transfer and the cooling rate. The simplest model describing the temperature geometry in the workpiece, which was derived by Rosenthal (1946), is given by

$$T(r) = \frac{Q}{2\pi\lambda}\frac{1}{r} + T_0 \qquad (1.1.1)$$

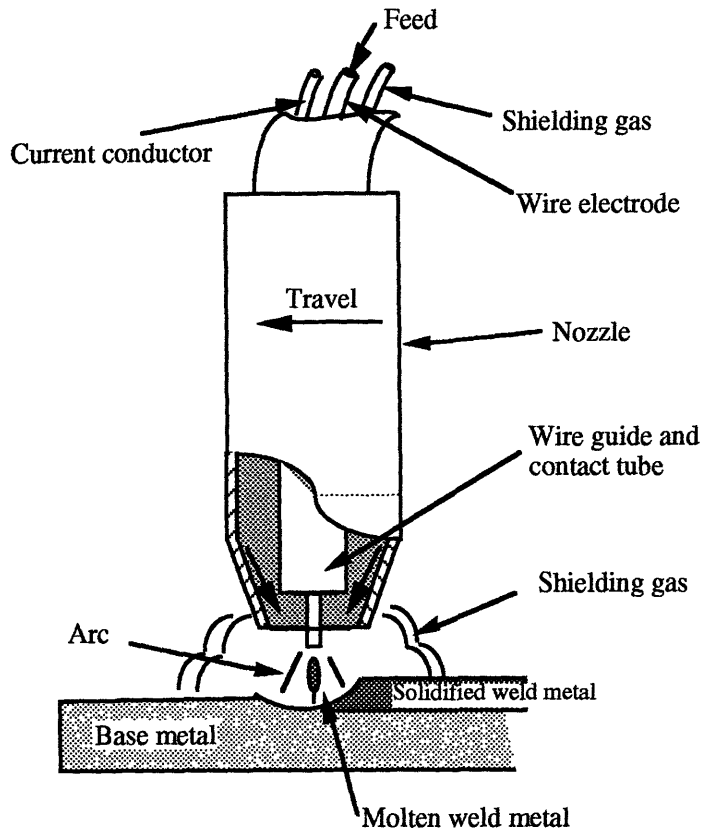


Fig. 1.1.1 GMAW

where T denotes the temperature, T_0 is the ambient temperature, Q is the heat input, λ is the thermal conductivity, and r is the distance from a torch. The equation (1.1.1) results in the model for the cooling rate:

$$CR = \frac{v}{Q} 2\pi\lambda (T_c - T_0)^2 \qquad (1.1.2)$$

where CR is the cooling rate at the critical temperature T_c, and v is the travel speed. The mass transfer is simply proportional to the ratio of the wire feed rate and the travel speed. Since almost all heat would be supplied by the molten electrode,

$$\text{Mass transfer} = \propto \frac{f}{v} \propto \frac{Q}{v} \qquad f\text{: Wire feed rate} \qquad (1.1.3)$$

is also true for pure arc process as well. Therefore, the mass transfer is closely coupled to the cooling rate via the term Q/v. From this point of view, Rosenthal's model shows that it is impossible to control the centerline cooling rate and the mass transfer at the same

time. However, as will be shown later, a new careful analysis and experimental data show that this is not the case.

Therefore, the first objective of this thesis is to depict the steady-state and dynamic behavior of the centerline cooling rate. Although extensive investigation has been done by Doumanidis and Hardt (1988, 1989) on the treatment of thermal features from a view-point of control scheme, little description for the detailed behavior of the centerline cooling rate has been discussed for these two inputs.

The other purpose of this thesis is to design the control system for the centerline cooling rate. As Doumanidis and Hardt (1988, 1989) pointed out, the control must deal with a nonlinearity in the dynamics of the cooling rate. In this thesis two types of controls, linear adaptive control and control using neural networks, are presented to deal with the nonlinearity.

The linear adaptive control is used as a system to adjust a linear model to the nonlinear dynamics of the process and to control it through the linear model. By 1980 considerable theoretical progress had been made in the scheme of adaptive control (Narendra and Annaswamy (1989), Åström (1989), Goodwin and Sin (1984) and Landau (1981)). The applications of the adaptive control to the welding process have been presented by Suzuki (1988), Song (1992), and Doumanidis (1988). These investigations show satisfactory results for the control of the welding process. From practical point of view, the implementation of an adaptive control is rather simple. However, the control sometimes shows an undesirable behavior such as a saturation and an instability due to the violation of the strong assumptions of most adaptive schemes. Therefore an appropriate consideration should be given to the implementation of the algorithm. In this thesis we focus our interest on the practical implementation of the control for the cooling rate.

Recently, control using neural networks has been attracting the interest of many researchers. (White et al. (1992)) The expectation for neural networks is that they will give us a tool designing a nonlinear controller. Many control schemes using neural networks have been proposed. However the theoretical treatments are still under research. Little successful application of control using neural networks has been reported. Neural networks simply provide a model describing nonlinear systems, therefore various nonlinear control schemes are applicable. In this thesis first the possible configuration of the control using neural networks for the cooling rate is studied, considering the limitations of neural networks and the nature of the process of the cooling rate. Then, the design methods and the performance of this control are discussed.

1.1 Organization of the thesis

Chapter 2 describes the experiment for the cooling rate that was conducted to show the effect of the inputs of wire feed rate and travel speed. Chapter 3 presents an analytical model for analyzing the dynamics of the cooling rate. This model is used for designing the controller for the cooling rate. Chapter 4 discusses the nature of the cooling rate process based on the results in Chapter 2 and Chapter 3. In this chapter the performance of conventional PI controller is shown to demonstrate the problems of the control for the cooling rate. Chapter 5 reviews nonlinear controls to study the feasible nonlinear control methods for the cooling rate. Chapter 6 deals with adaptive control for the cooling rate. In this chapter practical implementation issues are discussed. Chapter 7 presents control using neural networks. In this chapter the drawbacks of the neural networks are first investigated. Then we discuss the implementation, considering the drawbacks and the nature of the cooling rate dynamics. Finally conclusions are made in Chapter 8.

Chapter 2

Measurement of Cooling Rate during GMAW

It is quite important to know and understand the nature of the process when we design the controller. This chapter presents the measurement of the center-line cooling rate during GMAW for this purpose. The objective of the measurement is to learn how the input variables of wire feed rate and travel speed affect the cooling rate.

2.1 Measurement system

Some measurement systems have been presented for measuring thermal geometry during welding. (Doumanidis and Hardt (1988, 1989), Lukens and Morris (1982), and Chin, Madsen and Goodling (1983)) All the systems use an infrared camera, which usually require cooling by liquid nitrogen, and is a rather expensive sensor.

However, the purpose of our experiment is simply to measure the temperatures along a line on the weldment surface. Therefore, a simpler configuration of measurement system is proposed. It is composed of an infrared pyrometer that is widely used in the industry and an oscillating mirror.

2.1.1 Specifications of measurement system

Figure 2.1.1 shows the measurement system, whose specification is shown in Table 2.1. About 45 output signals proportional to the radiation from the weld surface behind the torch are acquired by a PC acquisition board every scanning of the mirror. The sampling timing is synchronized with the oscillation of the mirror by a pulse signal that is given by the step motor. (Fig. 2.1.2) The signals from the pyrometer are converted to temperatures by the method discussed in Section 2.1.2. The temperature distributions on the weldment are acquired every 0.5 sec. The acquired temperatures are fit to a spline function composed of piece-wise third-order polynomials to eliminate noise caused by the arc. The centerline cooling rate is calculated based on the polynomials.

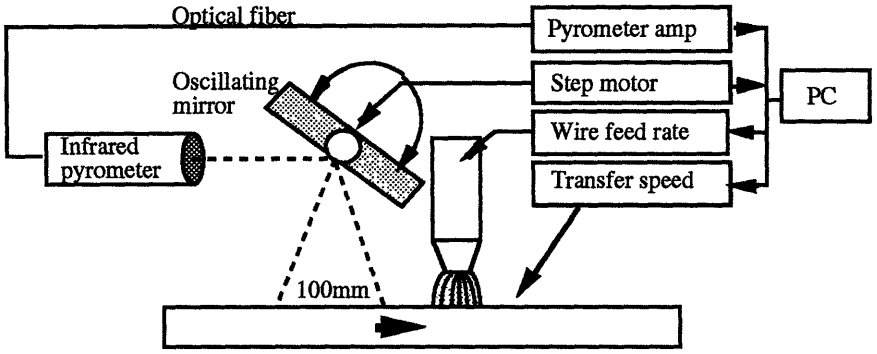


Fig. 2.1.1 Layout of measurement system

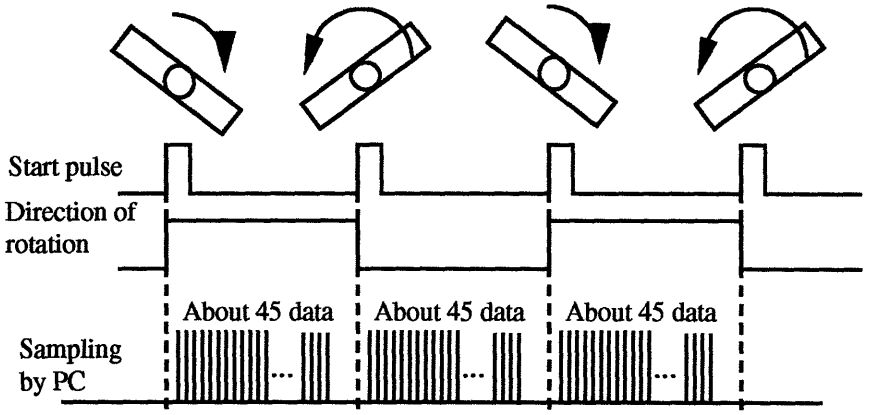


Fig. 2.1.2 Synchronization of sampling

Table 2.1 Specification of measurement system

Pyrometer	
Type	Accufiber model 100c (Accufiber Division Luxtron Corp.)
Photocell	Silicon cell
Wave length of pyrometer	950 nm
Output	Intensity proportional to radiation
Band width of output	10000 Hz

Oscillating mirror	
Oscillation cycle of mirror	1 Hz: Temperature distribution is measured every 500 msec
Location of mirror	Height: About 100 mm, Distance from arc: About 50 mm
Angle of oscillation	27 degree: Temperature distribution within about 100 mm from the arc is measured.

Data acquisition	
Data acquisition method	Data acquisition board in PC
Data acquisition interval	10 msec: About 45 data are sampled per a scanning of mirror

2.1.2 Conversion of intensity of radiation to temperature

(1) Plank equation

The relationship between radiated energy and temperature is given by the Plank equation:

$$E(T,\lambda) = \varepsilon \frac{c_1 \cdot \lambda^{-5}}{\exp(\frac{c_2}{\lambda T}) - 1}$$

$$c_1 = 3.743 * 10^8 \ [\mathrm{w} \cdot \mu\mathrm{m}^4 / \mathrm{m}^2] \qquad (2.1.1)$$

$$c_2 = 1.4387 * 10^4 \ [\mu\mathrm{m} \cdot \mathrm{k}]$$

where λ is the wavelength [μm], T denotes the temperature [k] and ε is the emissivity. The signal output of a pyrometer is proportional to the radiation energy. Therefore, equation (2.1.1) yields the following equation to calculate the temperature T.

$$T = \frac{c_2}{\lambda} \frac{1}{\ln(k\frac{\lambda^{-5}}{I} + 1)} \qquad (2.1.2)$$

$$k = \varepsilon \cdot c_1 \cdot c_3$$

where I is the signal output [v], c_3 is the coefficient for the transformation from the radiated energy to the signal output.

The coefficient k changes by the amplification of the internal circuit in a pyrometer. The value of k is determined by calibration using a blackbody (Emissivity = 1.0). The choice of the amplifier depends on the temperature range that we want to measure, considering the voltage that can be acquired by PC, which is from 0 to 10 v.

We will use the amplifier whose value of k is given as $5.27 * 10^5$ by the calibration. Then, the maximum value of the measurable temperature corresponding to the signal of 10 v is about 1100 °C, when we assume the emissivity is 0.8. The minimum temperature, which depends on the signal/noise ratio, is about 600 °C. It is noted that the accuracy of the measurement depends on the emissivity. However, we can not determine the value exactly, because it varies, depending on the material and the condition of the surface that is measured. We usually take 0.8 as the emissivity that is widely used for the measurement of steel temperature. We also note that the measurement discussed above is based on the following assumption.

1. The pyrometer has an optical filter with a narrow band around a specified wavelength 950 nm.

2. The photocell in the pyrometer yields the signal proportional to the radiation energy of the specified wavelength.
3. The emissivity has an almost constant value and can be predetermined.

(2) Comparison with temperature measured by thermocouple

We can measure temperature by a pyrometer under the assumptions shown above. However, we can not say whether these assumptions are true or not. It is reported that a pyrometry sensor brings about a satisfactory measurement for the temperature greater than 1000 °C. We do not know how it performs for the temperature range of our interest from 600 °C to 1000 °C. We also have a fundamental question for the emissivity, because it varies by various factors such as the surface condition and the material. Nobody can determine the value without comparing the output of the pyrometer with a temperature measured by other methods.

In order to eliminate the uncertainties of these assumptions, a comparison between the measurement by the pyrometer and that by a thermocouple is made. Figure 2.1.3 shows the results of the comparison. In the figure we take 0.8 as the emissivity. We can see a big difference between both measurements. The temperature by the pyrometer shows a higher value than that by the thermocouple. We can not explain the difference only by the error of assumed emissivity.

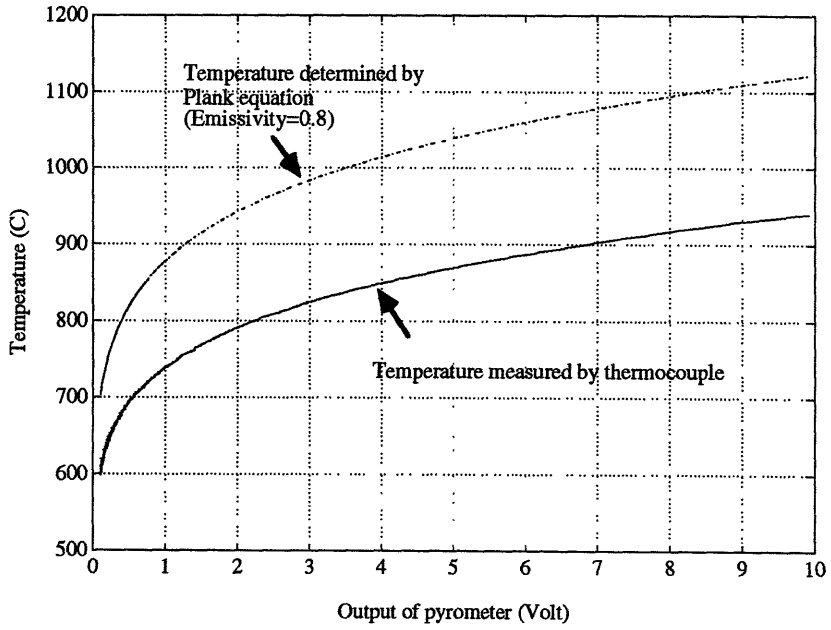


Fig. 2.1.3 Comparison between measurements by pyrometer and thermocouple

(3) Conversion from output signal of pyrometer to temperature

As shown above, the difference between two measurements is quite big. In general the measurement by a thermocouple is more reliable than that by a pyrometry sensor. Therefore, we give up using the equation (2.1.2) for the conversion from the output signal of the pyrometer to the temperature. Instead we will use directly the relation between the temperature by a thermocouple and the output signal, which is given by the curve in Fig. 2.1.3, as a reference table for the conversion.

2.1.3 Fitting by spline function

The calculation of cooling rate involves a derivative of the temperature distribution. Therefore a noisy temperature would give us a fluctuating measurement of the cooling rate. It is vital to remove as much noise as we can. Figure 2.1.4 shows an example of a measured temperature distribution of a weldment, where the noise is quite evident.

In order to eliminate the noise, temperatures are fit by a spline function composed of piece-wise third order polynomials.

The procedures for the fitting are as follows.

1. Temperatures are fit by a third order polynomial.
2. If the error of the fitting by the third order polynomial is greater than a specified value, the temperatures are fit by a spline function.

In order to reduce over-fitting, a third order polynomial is used as a basic function.

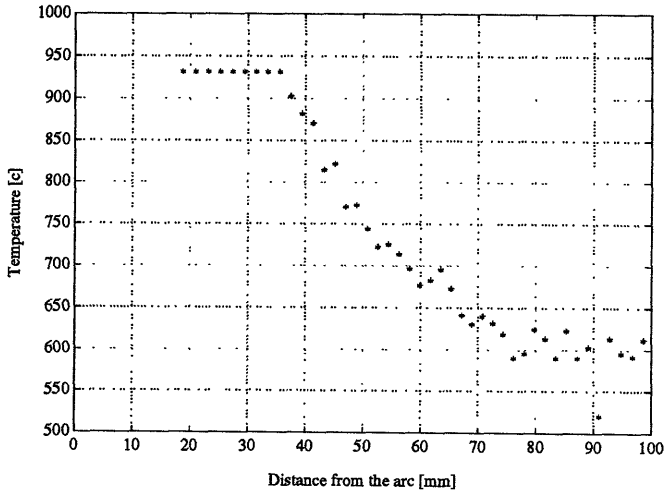


Fig. 2.1.4 Example of temperature distribution

(1) Fitting by third order polynomial

Figure 2.1.5 shows the fitting by a third order polynomial for the same data as shown in Fig. 2.1.4. In the figure the temperatures less than 600 °C are neglected, because they are so noisy that the fitting is aggravated. We can see the third order polynomial well fits the data.

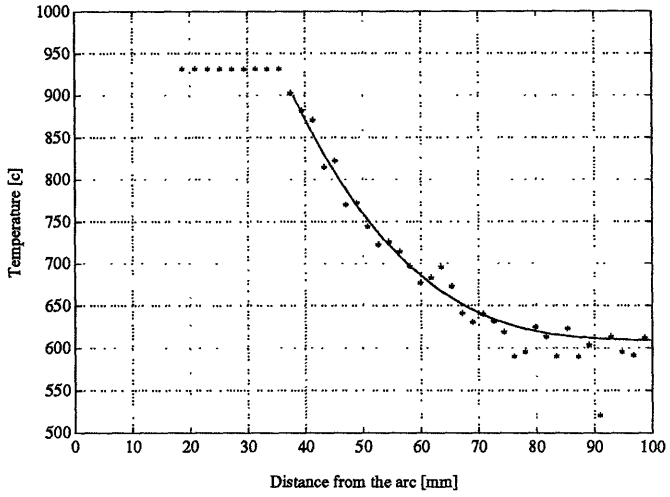


Fig. 2.1.5 Fitting by the 3rd order polynomial

(2) Fitting by spline function

The data shown in Fig. 2.1.4 are a typical steady-state temperature distribution. Steady-state temperatures can be fit well by a third order polynomial. However, if they are in the transient state it is sometimes hard to describe them as a third order polynomial. Figure 2.1.6 shows the fitting by a third order polynomial for a transient-state temperature distribution. We can see the third order polynomial is not enough to fit these data.

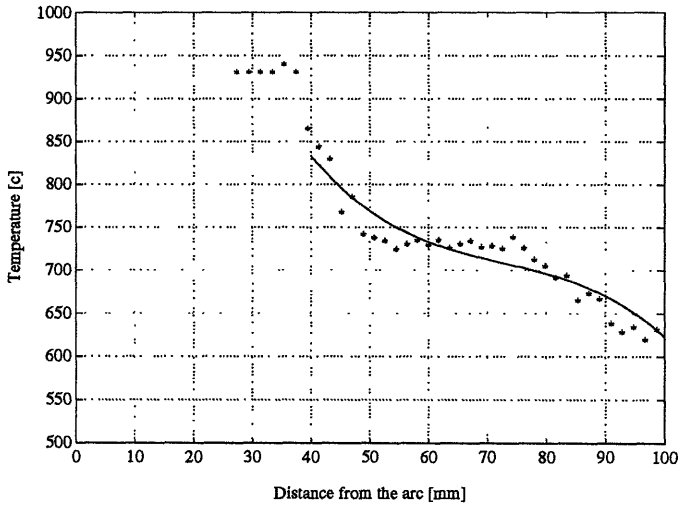


Fig. 2.1.6 Fitting for transient-state temperature distribution

Such a temperature distribution should be fit by a higher order polynomial. However, it is known that a high order polynomial sometimes causes some fluctuations in its curvature. Another function which is often used is a spline function. A spline function, a continuous function that is composed of some piece-wise polynomials, is given by

$$\begin{cases} f(x) = f_i(x) \quad (b_{i-1} < x \le b_i) \\ f_i^{(k)}(b_i) = f_{i+1}^{(k)}(b_i) \ (k = 0 \ \cdots \ (m-1)) \quad \text{(Continuity on nodes)} \end{cases} \tag{2.1.3}$$

where $f_i(x)$ is the m-th order polynomial given by $\sum_{j=0}^{m} a_{ij}x^j \ (i = 1 \cdots n)$, and b_i denotes the nodes, and then $b_0 = -\infty, b_n = \infty$.

It is known that the spline function can be rewritten as:

$$f(x) = \sum_{i=0}^{m} c_i x^i + \sum_{j=1}^{n-1} d_j (x - b_j)_+^m \tag{2.1.4}$$

$$x_+^m = \begin{cases} x^m \ (\mathrm{x} \ge 0) \\ 0 \quad (\mathrm{x} < 0) \end{cases} \quad \text{(Truncated m-th order polynomial)} \tag{2.1.5}$$

Therefore, the fitting by spline functions results in the estimation of the coefficient c_i and d_j, which can be determined by the similar manner as the fitting by polynomials. The fitting by spline functions has the following good points:

1. It does not bring about fluctuations that are sometimes produced by a high order polynomial.

2. It is continuously differentiable.

Figure 2.1.7 shows the fitting by a spline function, which is composed of three third order piece-wise polynomials. We can see that it represents the temperatures well.

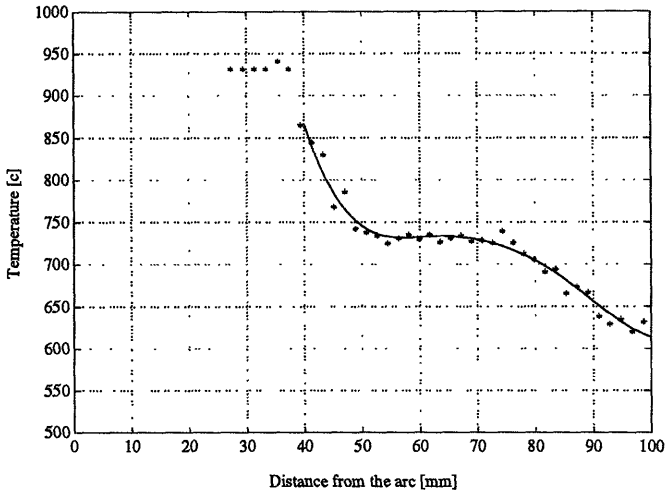


Fig. 2.1.7 Fitting by spline function

2.1.4 Other issues for measurement system

As shown in the figures above, the temperature is measured with respect to the location from the arc. Therefore, we need to know the location. The location is calculated geometrically from the angle of the mirror and the position where the mirror is situated. Since it is hard exactly to measure the position of the mirror, the relation between the angle of the mirror and the location where the pyrometer focuses is calibrated before experiment. To make the calibration, first a light source of a laser beam is mounted on the tube of the pyrometer. Then, the calibration data are obtained by observing the beam, and measuring the focused locations for some specified angles of the mirror.

To achieve a successful measurement of the temperature of the weldment, good alignment of the mirror and the pyrometer is crucial. In particular, the pyrometer should focus on the line of the weldment. We need check and adjust these devices by observing the laser beam so that the focal point will be located exactly on the line.

2.2 Measurement

Temperature gradient, cooling rate and location of critical temperature were observed during welding to see how the temperature distribution behaves for the step-wise change of travel speed and wire feed rate.

2.2.1 Measured variables

This section presents the definition of three variables, temperature gradient, cooling rate and location of critical temperature, and how to calculate them on-line. Before holding the discussion, we must keep in mind two types of coordinates for welding system. One is coordinates fixed to the workpiece, which will be called fixed coordinates. The other is coordinates fixed to the torch, called moving coordinates. The measurement is taken with respect to the moving coordinates. However, the definition of the cooling rate is based on the fixed coordinates. To avoid a confusion w is used as the location on the weldment with respect to the moving coordinates, and r denotes the location with respect to the fixed coordinates. $T(w,t)$ implies the temperature of the time t with respect to the moving coordinates, while $T'(r,t)$ is the temperature with respect of the fixed coordinates

Since $r = \int_{t_0}^{t} v(t)dt + r_0 + w$,

the relation between two representations of the temperature is given by

$$T(w,t) = T'(\int_{t_0}^{t} v(t)dt + r_0 + w, t) \qquad (2.2.1.a)$$

or $$T'(r,t) = T(-\int_{t_0}^{t} v(t)dt - r_0 + r, t) \qquad (2.2.1.b)$$

where r_0 denotes the location of the torch at the initial time t_0, and $v(t)$ is the velocity of the torch.

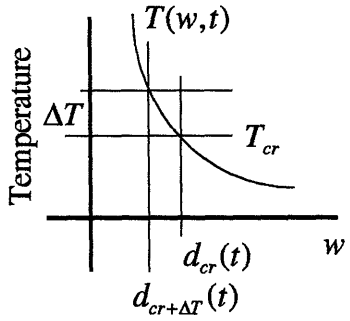
Temperature gradient

Temperature gradient Tg, the change of temperature per unit length at the critical temperature T_{cr}, is defined mathematically by

$$Tg(t) = -\left.\frac{\partial T'(r,t)}{\partial r}\right|_{T=T_{cr}} = -\left.\frac{\partial T(w,t)}{\partial w}\right|_{T=T_{cr}} \tag{2.2.2}$$

In the actual measurement, it is calculated by the following method:

$$\begin{aligned} Tg &= \frac{\Delta T}{d_{cr}(t) - d_{cr+\Delta T}(t)} \\ d_{cr}(t) &= \arg_{w}\{T(w,t) = T_{cr}\} \\ d_{cr+\Delta T}(t) &= \arg_{w}\{T(w,t) = (T_{cr} + \Delta T)\} \end{aligned} \tag{2.2.3}$$



where $d_{cr}(t)$ denotes the location of critical temperature. The temperature gradient represents the spatial temperature change around the critical temperature.

Cooling rate

Cooling rate Cr, the change of temperature per unit time at the critical temperature, is given by

$$Cr(t) = -\left.\frac{\partial T'(r,t)}{\partial t}\right|_{T=T_{cr}} \tag{2.2.4}$$

which is considered one of factors to represent the mechanical and metallurgical properties of the bead after welding. In steady-state this value is equivalent to the temperature gradient multiplied by the travel speed, because the temperature distribution in the moving coordinate is not dependent on time t, and the cooling rate is calculated as:

$$Cr = -\left[\frac{\partial w}{\partial t}\right]_{T=T_{cr}} \cdot \left[\frac{\partial T(w)}{\partial w}\right]_{T=T_{cr}} = v \cdot Tg \tag{2.2.5}$$

However, in the transient state it should be calculated as:

$$Cr = -\left[\frac{\partial w}{\partial t}\right]_{T=T_{cr}} \cdot \left[\frac{\partial T(w,t)}{\partial w}\right]_{T=T_{cr}} - \left[\frac{\partial T(w,t)}{\partial t}\right]_{T=T_{cr}} = v \cdot Tg - \left[\frac{\partial T(w,t)}{\partial t}\right]_{T=T_{cr}} \tag{2.2.6}$$

The temperature gradient multiplied by travel speed is sometimes used as an alternative definition of the cooling rate. However, this definition brings about an unrealistic output in the transient state. For example, if the travel speed is abruptly changed, the cooling rate would also change abruptly, which is physically not possible.

The difficulty in the calculation of the cooling rate comes from the fact it involves the time and spatial derivatives, as shown in (2.2.6). Although various methods can be considered, the following will be used:

$$CR(t) = \arg\min_{a} \sum_{i=0}^{N} (\Delta T_i - a \cdot i \cdot \Delta t - b)^2 \qquad (2.2.7)$$

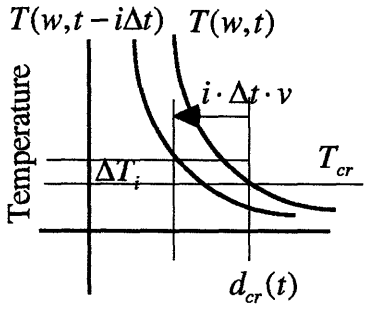
(Line – approximation)

$$\Delta T_i = T(d_i', t_i') - T_{cr} \qquad (2.2.7.a)$$

$$t_i' = t - i \cdot \Delta t \qquad (2.2.7.b)$$

$$d_i' = d_{cr}(t) - i \cdot \Delta t \cdot v \qquad (2.2.7.c)$$

$$d_{cr}(t) = \arg_{w}\{T(w,t) = T_{cr}\} \qquad (2.2.7.d)$$



where v denotes the travel speed, and Δt is the sample interval. t_i' is the time when the past sample data was taken. In (2.2.7.c), d_i' implies where the location that presently has the critical temperature was located in the past temperature distribution $T(w,t_i')$. Therefore, ΔT_i shows how much the temperature drops from the time t_i' to the time t and reaches the critical temperature at the time t.

As shown above, we must use the present and past temperature distributions, which are usually contaminated by noise even if the temperature distributions are fit by spline functions to eliminate the noise. Therefore, if we calculate the cooling rate simply by $\Delta T_1/\Delta t$, the resultant data are quite fluctuating. To avoid this, the approximation shown in (2.2.7) is used.

Since the temperature drop ΔT_i is not proportional to the time difference $(i \cdot \Delta t)$, it is noted the measured cooling rate is not necessarily equal to the measured temperature gradient multiplied by velocity even in the steady-state due to the difference in the calculation methods of temperature gradient and cooling rate.

2.2.2 Measurement conditions

The variables above are measured for the following 4 cases:

Case A: Travel speed is kept constant at 4 mm/sec. Wire feed rate is changed from 18 cm/sec to 25 cm/sec as a step input.

Case B: Travel speed is kept constant at 8 mm/sec. Wire feed rate is changed from 18 cm/sec to 25 cm/sec as a step input.

Case C: Wire feed rate is kept constant at 18 cm/sec. Travel speed is changed from 4 mm/sec to 8 mm/sec as a step input.

Case D: Wire feed rate is kept constant at 25 cm/sec. Travel speed is changed from 4 mm/sec to 8 mm/sec as a step input.

The experiments were executed on mild steel, 12.7 mm thick and 60 mm in width under welding conditions of open-circuit voltage 30 v, Argon-2% oxygen shielding gas flowing at 100 l/min, and welding wire diameter 0.035 in.. The critical temperature is taken as 720 °C, given that the austenitization temperature is 727 °C.

2.2.3. Result of measurement

(1) The case where the travel speed is kept constant

Case A

Figure 2.2.1 shows the temperature gradient, the location of critical temperature, and the cooling rate for the case A, where the travel speed is 4 mm/sec and the wire feed rate is changed from 18 mm/sec to 25 mm/sec. The following results can be observed from this figure.

Temperature gradient

The temperature gradient gradually goes up form 9 °C/mm to 15 °C/mm after the wire feed rate changes from 18 mm/sec to 25 mm/sec. After it reaches 15 mm/sec, it gradually goes down to the original point of 9 °C/mm.

Location of critical temperature

The location of critical temperature gradually moves away from the arc after the step input is given. It moves from 46 mm to 55 mm. After it reaches 55 mm, it is kept at this value.

Cooling rate

The cooling rate remains constant at about 40 °C/sec even if the wire feed rate is changed, although a distinct change in the temperature distribution can be seen from the responses of the temperature gradient and the location of critical temperature.

Some transient variation of the cooling rate after the step-wise change of the wire feed rate may take place, but be hidden in the noisy data. However, it is important to see that there would be no difference in the steady-state cooling rate before and after the step input. This result is also verified by the figure of the temperature gradient because the cooling rate in the steady-state is calculated as the temperature gradient multiplied by the travel speed. As mentioned above, the temperature gradient has the same value of 9 °C/mm in the steady-state before and after the step-wise change of the wire feed rate.

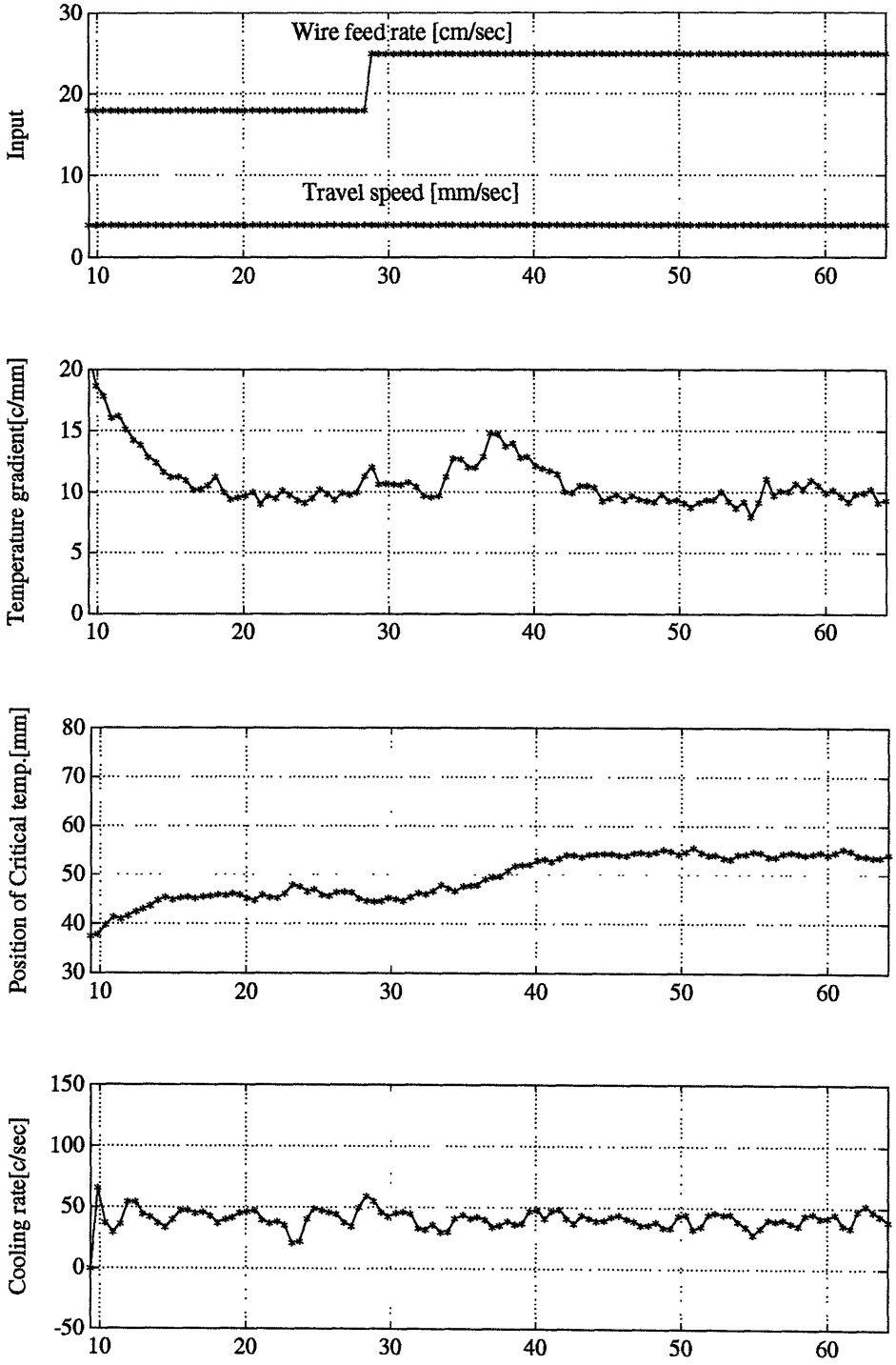


Fig. 2.2.1 Result of measurement for case A

Case B

Figure 2.2.2 shows the results for the case B, where the travel speed is 8 mm/sec and the wire feed rate is changed from 18 mm/sec to 25 mm/sec. The following results similar to the case A are obtained from the figure.

Temperature gradient

It gradually goes up from 11 °C/mm to 15 °C/mm after the step-wise change of the wire feed rate. After it reaches the peek, it gradually goes down to the original point of 11 °C/mm.

Location of critical temperature

The location of critical temperatures gradually moves away from the arc after the step input. It moves from 47 mm to 53 mm. After it reaches 53 mm, it keeps the value.

Cooling rate

There is no distinct change in the cooling rate. It is kept around 130 °C/sec.

Comments on the cases where the travel speed is kept constant

1. It is clear that the heat input per unit time grows by the increase of the wire feed rate because the location of the critical temperature moves away from the arc after the increase.
2. The obvious increase of the width and height of the bead after the step input was observed in the experiments.
3. However, the cooling rate remains at almost same value.
4. Under the assumption that the heat input is proportional to the wire feed rate and the temperature is subject to the Rosenthal's model, the cooling rate should show about 38% increase.$((25-18)/18=38\%)$ However, the experiments show that is not the case.

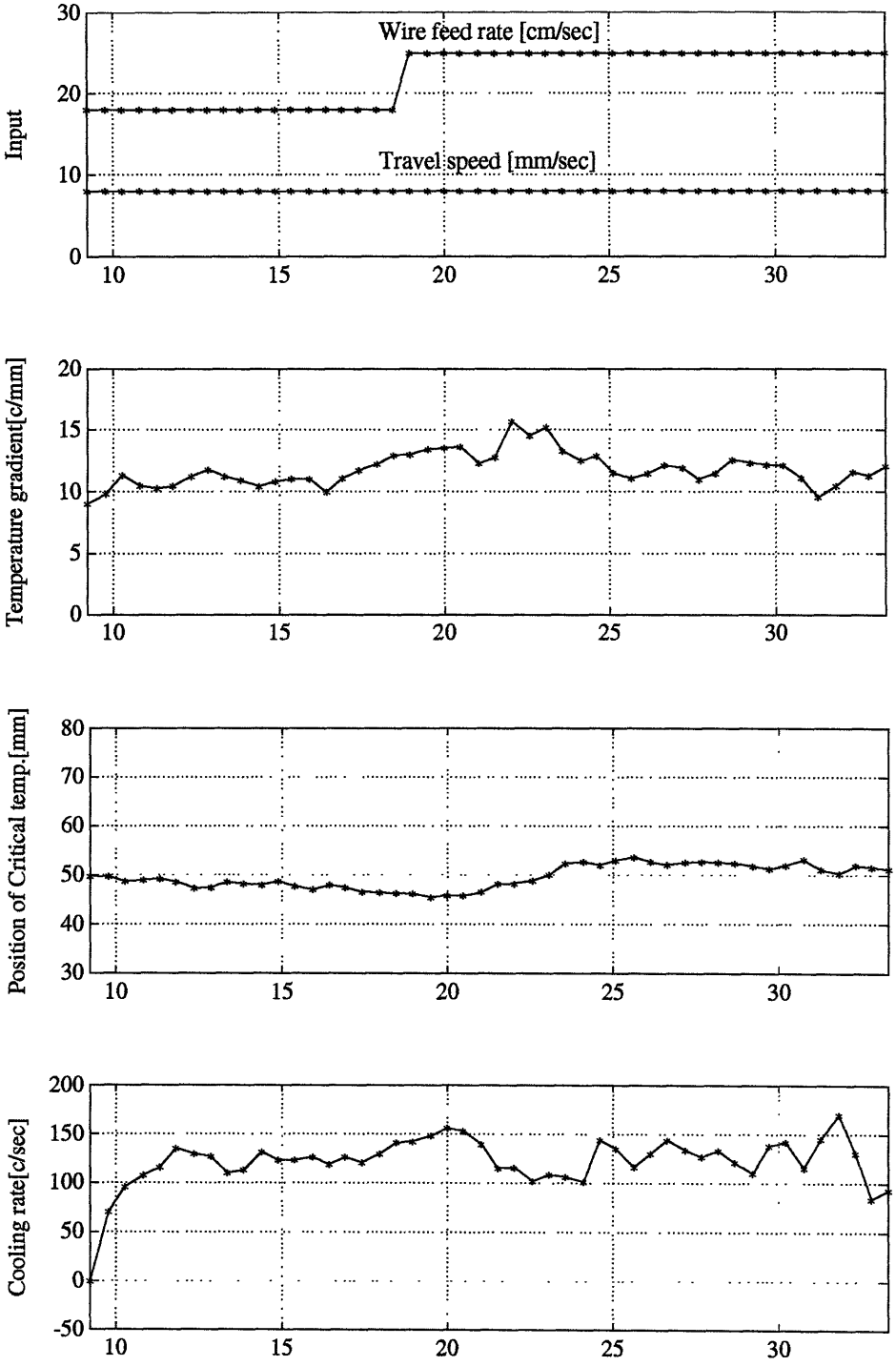


Fig. 2.2.2 Result of measurement for case B

(2) The case where the wire feed rate is kept constant

Case C

Figure 2.2.3 shows the results for the case C, where the wire feed rate is 18 cm/sec and the travel speed is changed from 4 mm/sec to 8 mm/sec. The results are distinctly different from the case A and B.

Temperature gradient

The temperature gradient remains almost constant for a while after the stepwise change of the travel speed. It shows a sharp valley in its plot at almost same time when the location of the critical temperature shows a significant drop. The temperature gradient in the steady-state shows a little change from 9 °C/mm to 11 °C/mm. This implies the steady-state cooling rate is almost proportional to the travel speed.

Location of critical temperature

The location of critical temperature shows a gradual increase from 48 mm to 65 mm after the change of the travel speed. When it reaches the peak, it suddenly decreases to the value of 45 mm and keeps the value. This sudden drop is quite different from the case of stepwise change of wire feed rate.

Cooling rate

The cooling rate does not show any significant variation for a while since we changed the travel speed. However, it suddenly increases in value from 40 °C/sec to 110 °C/sec 6 seconds later after the stepwise change. It shows a response involving a quite long delay and then a first order response with a short time constant.

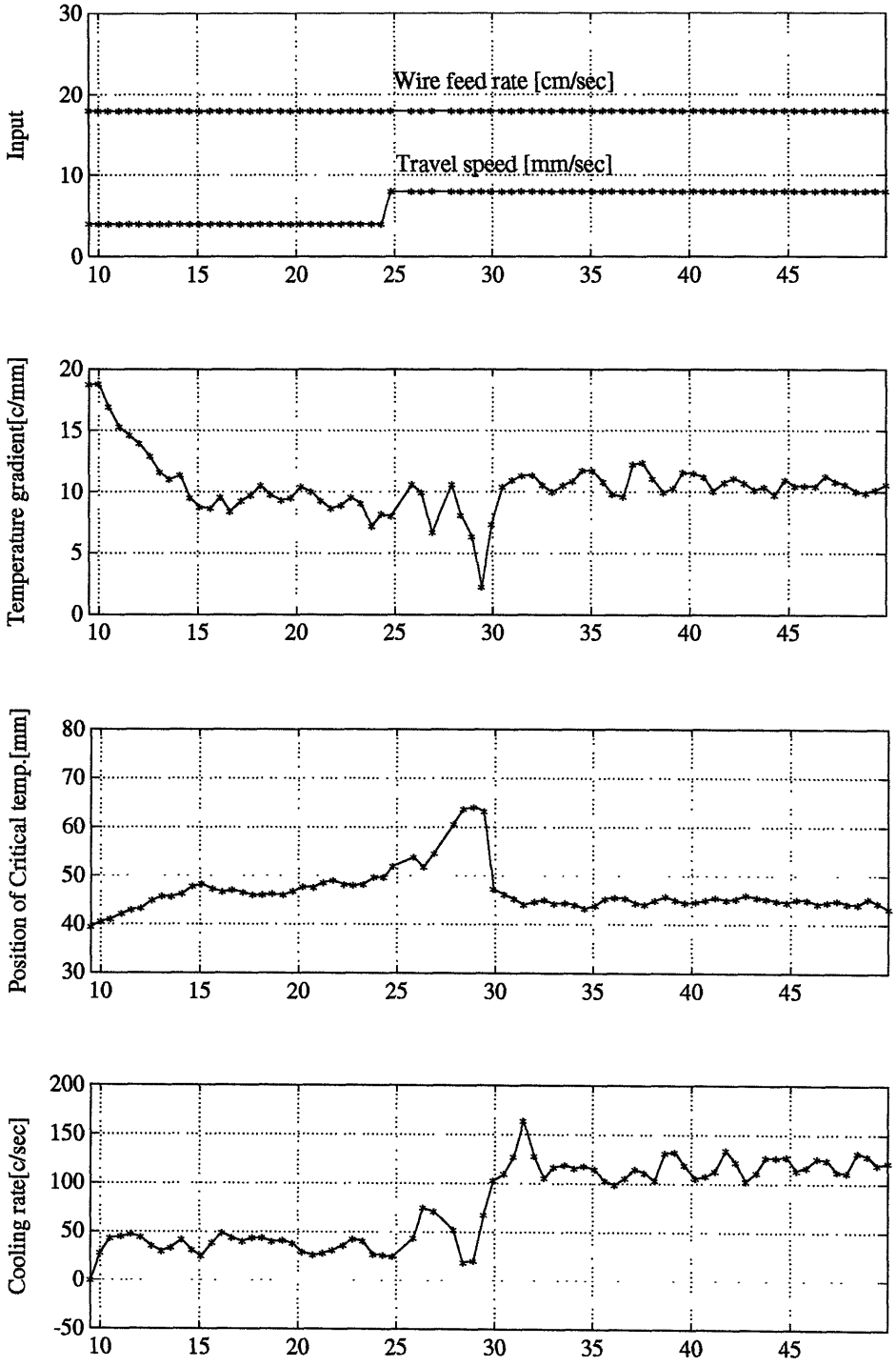


Fig. 2.2.3 Result of measurement for case C

Case D

Figure 2.2.4 shows the results for the case D, where the wire feed rate is 25 cm/sec and the travel speed is changed from 4 mm/sec to 8 mm/sec. The similar results to the case C are obtained.

Temperature gradient

Temperature gradient keeps the same value for about 5 seconds after the stepwise change of the travel speed. It shows a sharp valley in its plot. The temperature gradient in the steady-state shows a change from 8 °C/mm to 12 °C/mm.

Location of critical temperature

The location of critical temperature shows the gradual increase from 55 mm to 75 mm after the change of the travel speed. When it reaches the peak, it suddenly drops to the value of 49 mm and keeps the value.

Cooling rate

The cooling rate suddenly increases its value from 40°C/sec to 120 °C/sec 6 seconds later after the step input.

Comments on the case where the wire feed rate is kept constant

1. The heat input per unit time would be constant, because the wire feed rate is not changed.
2. The distinct decrease of the width and height of the weldment after the step input was observed in the experiments.
3. The step-wise change of the travel speed produces a time-delay and a first order response with a short time constant for the cooling rate.

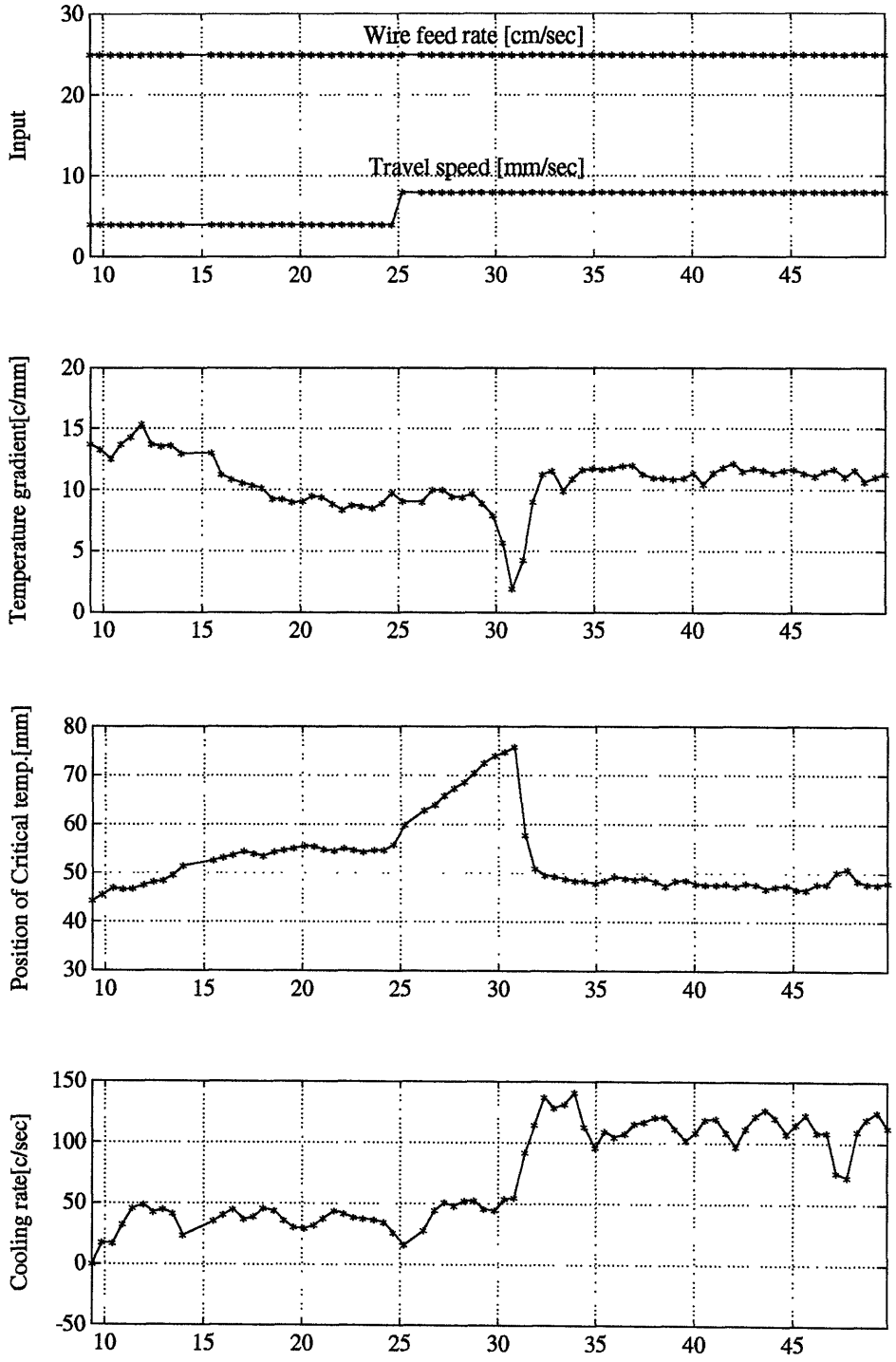


Fig. 2.2.4 Result of measurement for case D

2.3. Remarks

1. Figure 2.2.5 shows the typical patterns of measured variables. First, these patterns show a distinct difference between the case where the wire feed rate is changed, and the case where the travel speed is changed. Of particular interest is the observation that the wire feed rate has little effect on the cooling rate, although a distinct change of the temperature distribution is observed. By contrast, the travel speed has a strong effect on cooling rate. This result suggests the cooling rate should be controlled by the travel speed, and can be controlled independently of mass transfer.

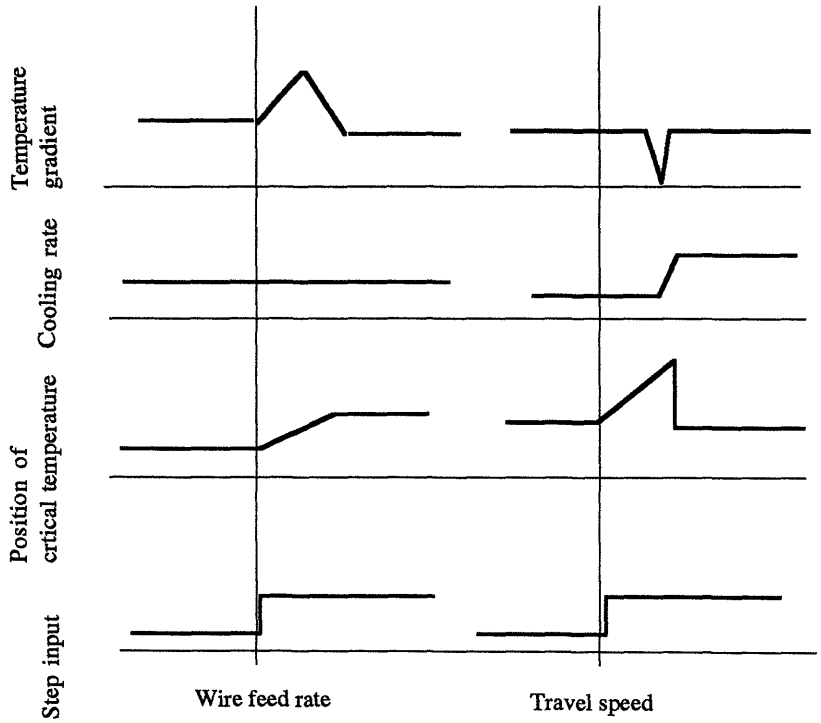


Fig. 2.2.5 Typical pattern of measured variables

2. Figure 2.2.6 shows the transient change of every one second in the temperature distribution for the case shown in Fig. 2.2.1, where the wire feed rate is changed from 18 cm/sec to 25 cm/sec. It performs as follows

a. After the step-wise change, the gradient of the temperature distribution becomes steeper from higher temperature. (① →②)

b. After a while, the gradient is decreasing from lower temperature. (② → ③)

c. The final shape of temperature distribution, the steady-state shape, is almost same as the initial. It looks like as if the temperature just shifted about 10 mm farther from the arc. That is the reason the steady-state cooling rate does not change even if the wire feed rate increases

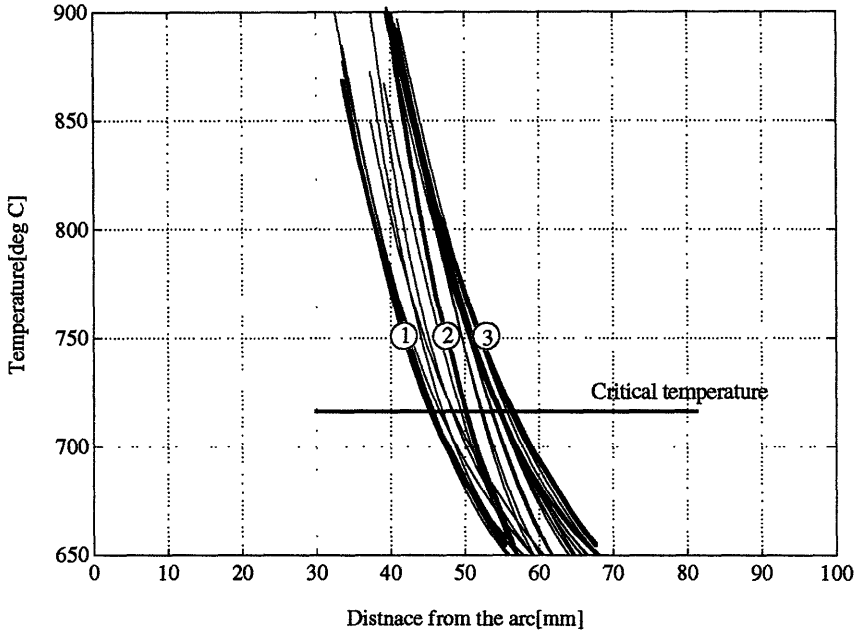


Fig. 2.2.6 Change in temperature distribution by wire feed rate

3. To see more clearly the results of the steady-state cooling rate, let us consider the condition in which the cooling rate is not affected by wire feed rate, or heat input.

According to the principle of superposition, the steady-state temperature $T(Q,w)$ can be written as

$$T(Q,w) = Qg(w) + T_0 \tag{2.3.1}$$

where Q is the heat input, and w is the distance from the arc, and T_0 are a constant. Then, the temperature gradient Tg is given by

$$Tg(Q,w) = -Q\,dg(w)/dw\big|_{T=T_{cr}} \tag{2.3.2}$$

Since we are concerned with the critical temperature, $T(Q,w) = T_{cr}$ (2.3.3)

Then,

$$(dQ/dw)g(w)\big|_{T=T_{cr}} = -Qdg(w)/dw\big|_{T=T_{cr}} \tag{2.3.4}$$

where dQ/dw shows the ratio between the change of the heat input and the movement of the location of the critical temperature which is caused by the heat input change.

We also have

$$\begin{aligned} \partial Tg(Q,w)/\partial Q &= -[dg(w)/dw + Q(d^2g(w)/dw^2)(dw/dQ)]\big|_{T=T_{cr}} \\ &= -[dg(w)/dw - (d^2g(w)/dw^2)g(w)/(dg(w)/dw)]\big|_{T=T_{cr}} \end{aligned} \tag{2.3.5}$$

In order that the steady-state cooling rate is not affected by the heat input,

$$\partial Tg(Q,w)/\partial Q = 0 \tag{2.3.6}$$

We can, therefore, get the condition:

$$(dg(w)/dw)^2 = (d^2g(w)/dw^2)g(w) \tag{2.3.7}$$

, which results in the solution

$$g(w) = A\exp(aw) \tag{2.3.8}$$

Therefore,

$$T(Q,w) = QA\exp(aw) + T_0 \tag{2.3.9}$$

More generally, we can write it as

$$T(Q,w) = f(Q)\exp(aw) + T_0$$

by the similar discussion.

In Fig. 2.2.7 the steady-state temperature distributions obtained by the experiments are fit by an exponential function (2.3.9). It describes well the steady-state temperature distribution for the measurement range of 600 °C to 900 °C. The condition that the cooling rate is not affected by the heat input is satisfied.

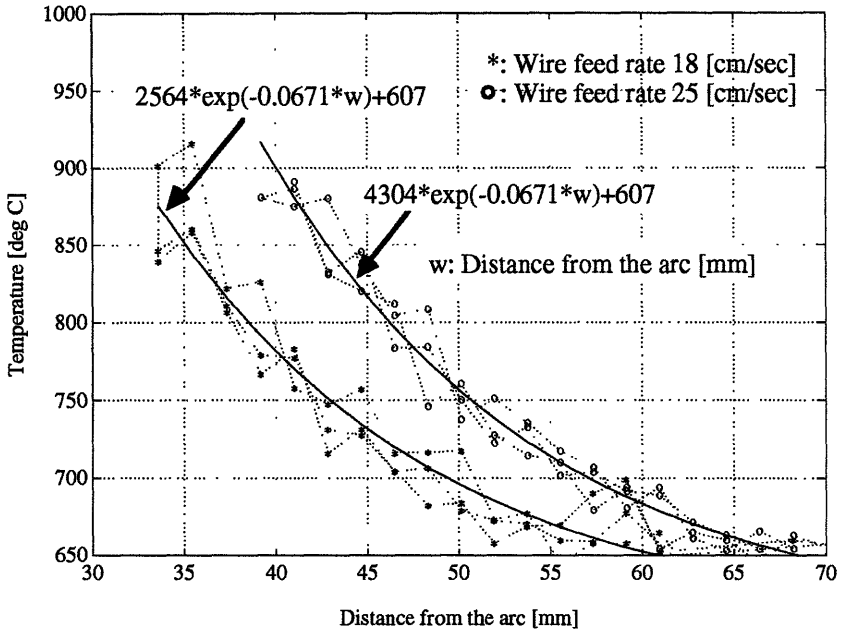


Fig. 2.2.7 Steady-state temperature distribution
(Travel speed 4 mm/sec)

4. Figure 2.2.8 shows the change of every 500 msec in the temperature distribution for the case shown in Fig. 2.2.4, where the travel speed is changed from 4 cm/sec to 8 cm/sec. The figure shows the following interesting characteristics.

a. After the step-wise change, the temperature distribution is shifting farther from the arc. (① →②)

b. After a while, it is decreasing its value from higher temperature. (② → ③ → ④)

c. The change from the state ③ to the state ④ is quite fast. That yields the sudden change of the temperature gradient, the location of the critical temperature and the cooling rate at the critical temperature.

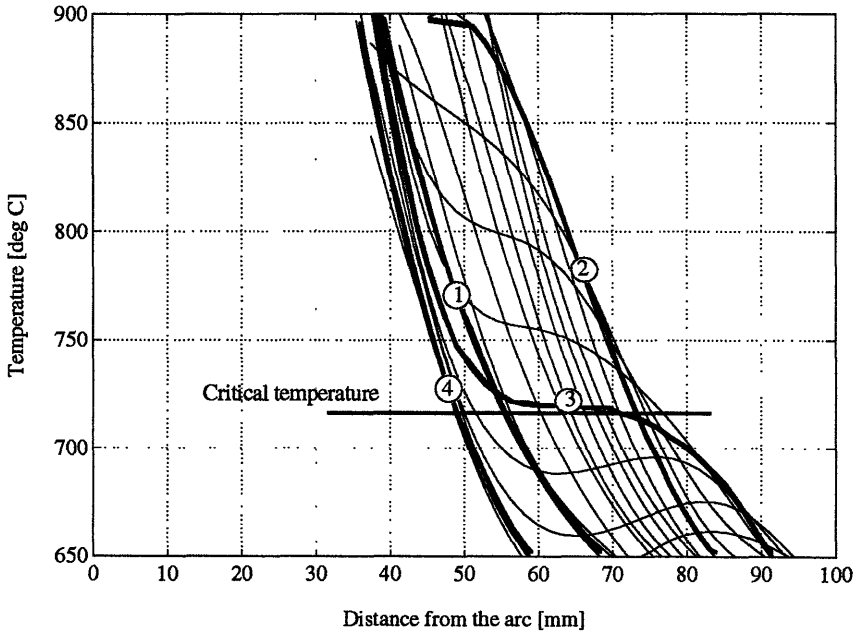


Fig. 2.2.8 Change in temperature distribution by travel speed

5. The cooling rate must be determined in accordance with its metallurgical definition. The temperature gradient multiplied by the travel speed is sometimes used as an alternative. However, during the transient-state, its value is fairy different from the actual one, which would badly mislead the control system.

Chapter 3

Dynamic Model of Temperature Distribution

When we apply control to a process, we always have to evaluate the performance. The evaluation sometimes requires quite many experiments and adjustments of the parameters of the control. A model that describes the dynamic behavior of the process is used to reduce such laborious experiments and tuning of the parameters. It also gives us insight for designing the controller. In this chapter we study a model for this purpose. In particular we turn our attention to analytical models based on the heat conduction partial differential equation.

(Nomenclature)

The following nomenclature will be used throughout the chapter.

$T(w,y,z,t)$	= Temperature [°C]
T_0	= Initial temperature, Ambient temperature [°C]
k	= Thermal diffusivity [m^2/hour]
λ	= Heat conductivity [kcal/(hour m °C)]
t	= Time [hour]
c	= Specific heat [kcal/kg]
ρ	= Density [kg/m^3]
h	= Scaled heat transfer coefficient [1/m]
	= Heat transfer coefficient[kcal/(hour m^2 °C)] divided by heat conductivity [kcal/(hour m °C)]
$q(\tau), Q$	= Heat input [kcal/hour] at time τ, and constant heat input
$v(t)$	= Travel speed [m/hour]
(w,y,z)	= Location in the moving coordinate
(x,y,z)	= Location in the fixed coordinate
a	= Thickness [m]
b	= Width [m]

3.1 Analytical models

The welding is a quite complicated process, which involves various phenomena such as heat conduction, heat convection, radiation, and mass transfer. The most important factor in the process is the heat conduction. However, we can not always neglect the other factors when we analyze the system. (Heiple et al. (1990)) Our concern lies in the cooling rate, which is a behavior after the solidification. Therefore, we focus our interest on heat conduction behavior.

Two types of models, an analytical model and a numerical model can be considered. However, the numerical model requires a large number of computations. That does not fit our purpose because we would like to evaluate the performance of the resulting controlled system by combining a controller with the model of the process. Therefore we use analytical models because they require fewer computations than numerical models.

The analytical models under various given conditions are shown in the sequel. We can derive these models by using Green's function methods (Appendix A). Extensive derivation of the analytical models was done by Rosenthal (1946). Rosenthal's models have assumptions such as a point heat source and the neglect of a heat flux from the surface. Also these models only deal with a steady-state solution. Therefore, we can not see how the temperature behaves while the heat input and the travel speed are changing dynamically. For this problem, a Green's function method is very helpful. The first derivation of a model using Green's functions was by Edgar and Tsai (1983) to get a more general model under the assumption the heat has a Gaussian distribution. However, it is still a steady-state model.

(1) Semi-infinite steady-state model (Rosenthal)

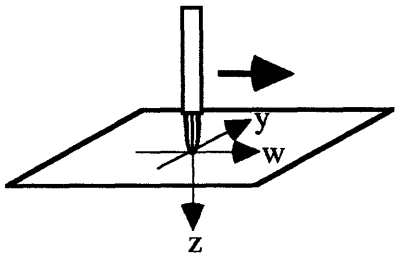
Conditions

(a) Semi-infinite space

(b) Point heat source

(c) No heat flux from the surface

(d) Steady-state model



Model

$$T(w,y,z) = \frac{Q}{2\pi\lambda}\frac{1}{r}\exp(-\frac{v}{2k}r)\exp(-\frac{v}{2k}w) + T_0 \qquad (3.1.1)$$

$$r = \sqrt{w^2 + y^2 + z^2}$$

(2) Model 1: Transient-state model under the same conditions as Rosenthal's model

Conditions

(a) Semi-infinite space

(b) Point heat source

(c) No heat flux from the surface

Model

$$T(w,y,z,t) = \int_0^t d\tau[\frac{1}{4}\cdot(\frac{1}{\sqrt{\pi}\sqrt{k}}\frac{1}{\sqrt{t-\tau}})^3\frac{q(\tau)}{c\rho}\exp(-\frac{(w+\int_\tau^t v(\eta)d\eta)^2 + y^2 + z^2}{4k(t-\tau)}) + T_0 \qquad (3.1.2)$$

If the heat input and the travel speed are kept constant at Q and v respectively, then

$$T(w,y,z,t) =$$

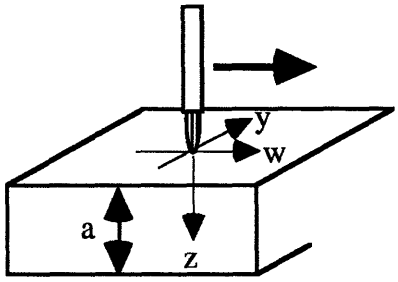
$$\frac{1}{4(\sqrt{\pi k})^3}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\int_0^t \frac{1}{(\sqrt{t-\tau})^3}\exp\left[-\frac{w^2+y^2+z^2}{4k}\frac{1}{t-\tau} - \frac{v^2}{4k}(t-\tau)\right]d\tau + T_0$$

$$= \frac{1}{4(\sqrt{\pi k})^3}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\int_{\frac{1}{\sqrt{t}}}^{\infty}\exp\left[-\frac{w^2+y^2+z^2}{4k}s^2 - \frac{v^2}{4k}\frac{1}{s^2}\right]ds + T_0 \quad (s = \frac{1}{\sqrt{t-\tau}}) \qquad (3.1.3)$$

We can see that

$\lim_{t\to\infty} T(w,y,z,t) \rightarrow$ Rosenthal's model

(3) Model 2



Conditions

(a) Finite thickness, and infinite width and length

(b) Point heat source

(c) Heat flux is described by:

$$\left[\frac{\partial T(w,y,z,t)}{\partial z}\right]_{z=0} = h\cdot(T(w,y,0,t)-T_0)$$

$$\left[\frac{\partial T(w,y,z,t)}{\partial z}\right]_{z=a} = -h\cdot(T(w,y,a,t)-T_0)$$

(3.1.4)

Transient-state model

$T(w,y,z,t)-T_0 =$

$$\frac{1}{4\pi k}\sum_{n=1}^{\infty}\int_0^t d\tau\frac{1}{t-\tau}\frac{q(\tau)}{c\rho}\exp\left(-\frac{(w+\int_\tau^t v(\eta)d\eta)^2+y^2}{4k(t-\tau)}-k\alpha_n{}^2(t-\tau)\right)\phi_n(z)\phi_n(0) \quad (3.1.5)$$

If the heat input and the travel speed are kept constant at Q and v, then

$$T(w,y,z,t)-T_0 =$$

$$\frac{1}{4\pi k}\frac{Q}{c\rho}\exp\left(-\frac{wv}{2k}\right)\sum_{n=1}^{\infty}\int_0^t\frac{1}{t-\tau}\exp\left[-\frac{w^2+y^2}{4k}\frac{1}{t-\tau}-\left(\frac{v^2}{4k}+k\alpha_n{}^2\right)(t-\tau)\right]d\tau\Phi_n(z)\Phi_n(0)$$

$$=\frac{1}{2\pi k}\frac{Q}{c\rho}\exp\left(-\frac{wv}{2k}\right)\sum_{n=1}^{\infty}\int_{\frac{1}{\sqrt{t}}}^{\infty}\frac{1}{s}\exp\left[-\frac{w^2+y^2}{4k}s^2-\left(\frac{v^2}{4k}+k\alpha_n{}^2\right)\frac{1}{s^2}\right]ds\Phi_n(z)\Phi_n(0)$$

(3.1.6)

where

$$\Phi_n(z) = \sqrt{\frac{2}{a + \frac{2\cos^2(\beta_n)}{h}}} \sin(\alpha_n z + \beta_n) \tag{3.1.7.a}$$

$$\alpha_n a + 2\beta_n = n\pi \tag{3.1.7.b}$$

$$\tan(\beta_n) = \frac{\alpha_n}{h} \tag{3.1.7.c}$$

Steady-state model

$T(w,y,z,\infty) - T_0 =$

$$\frac{1}{2\pi k}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\sum_{n=1}^{\infty}\int_0^{\infty}\frac{1}{s}\exp\left[-\frac{w^2+y^2}{4k}s^2 - (\frac{v^2}{4k} + k\alpha_n{}^2)\frac{1}{s^2}\right]ds\Phi_n(z)\Phi_n(0)$$

$$= \frac{1}{\pi k}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\sum_{n=1}^{\infty}K_0(\frac{\sqrt{v^2+4k^2\alpha_n{}^2}}{2k}\sqrt{w^2+y^2})\Phi_n(z)\Phi_n(0) \tag{3.1.8}$$

$K_0()$:The modified Bessel function of the second kind and zero order

(4) Model 3

Conditions

(a) Finite thickness, and infinite width and length

(b) Gaussian heat source

$$Q(x,y,z) = \frac{q}{c\rho}\frac{1}{2\pi\sigma^2}\exp(-\frac{w^2+y^2}{2\sigma^2})\delta(z) \tag{3.1.9}$$

(c) Heat flux is given by the equation (3.1.4).

Transient-state model

$T(w,y,z,t) - T_0 =$

$$\frac{1}{\pi}\sum_{n=1}^{\infty}\int_0^t \frac{1}{4k(t-\tau)+2\sigma^2}\frac{q(\tau)}{c\rho}\exp\left[-\frac{(w+\int_\tau^t v(\eta)d\eta)^2+y^2}{4k(t-\tau)+2\sigma^2} - k\alpha_n{}^2(t-\tau)\right]d\tau\Phi_n(z)\Phi_n(0) \tag{3.1.10}$$

If the heat input and travel speed are kept constant at Q and v, then

$$T(w,y,z,t)-T_0 =$$

$$\frac{Q}{\pi c\rho}\sum_{n=1}^{\infty}\int_0^t \frac{1}{4k(t-\tau)+2\sigma^2}\exp\left[-\frac{(w+v(t-\tau))^2+y^2}{4k(t-\tau)+2\sigma^2}-k\alpha_n{}^2(t-\tau)\right]d\tau\Phi_n(z)\Phi_n(0) \tag{3.1.11}$$

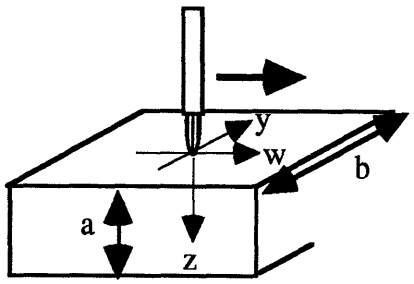
Steady-state model

We can not make the equation simpler for the steady-state. We just make time t infinite.

$$T(w,y,z,\infty)-T_0 =$$

$$\frac{Q}{\pi c\rho}\sum_{n=1}^{\infty}\int_0^{\infty} \frac{1}{4k(t-\tau)+2\sigma^2}\exp\left[-\frac{(w+v(t-\tau))^2+y^2}{4k(t-\tau)+2\sigma^2}-k\alpha_n{}^2(t-\tau)\right]d\tau\Phi_n(z)\Phi_n(0) \tag{3.1.12}$$

(5) Model 4



Conditions

(a) Finite thickness and width, and infinite length

(b) Point heat source

(c) Heat flux is described by :

$$\left[\frac{\partial T(w,y,z,t)}{\partial z}\right]_{z=0} = h\cdot(T(w,y,0,t)-T_0)$$

$$\left[\frac{\partial T(w,y,z,t)}{\partial z}\right]_{z=a} = -h\cdot(T(w,y,a,t)-T_0)$$

$$\left[\frac{\partial T(w,y,z,t)}{\partial z}\right]_{y=\frac{b}{2}} = 0 \tag{3.1.13}$$

$$\left[\frac{\partial T(w,y,z,t)}{\partial z}\right]_{y=-\frac{b}{2}} = 0$$

Transient-state model

$$T(w,y,z,t)-T_0 =$$

$$\frac{1}{4\sqrt{\pi k}}\sum_{\substack{n=1\\m=0}}^{\infty}\int_0^t d\tau\frac{1}{\sqrt{t-\tau}}\frac{q(\tau)}{c\rho}\exp\left(-\frac{(w+\int_\tau^t v(\sigma)d\sigma)^2}{4k(t-\tau)}-k\left(\alpha_n{}^2+\left(\frac{m\pi}{b}\right)^2\right)(t-\tau)\right)$$

$$\cdot\phi_n(z)\phi_n(0)\varphi_m(y)\varphi_m(0) \tag{3.1.14}$$

where

$$\Phi_n(z)=\sqrt{\frac{2}{a+\frac{a\cos^2(\beta_n)}{h}}}\sin(\alpha_n z+\beta_n) \tag{3.1.15.a}$$

$$\alpha_n a+2\beta_n=2n\pi \tag{3.1.15.b}$$

$$\tan(\beta_n)=\frac{\alpha_n}{h} \tag{3.1.15.c}$$

$$\Psi_m(y)=\begin{cases}\sqrt{\frac{2}{b/2}}\cos(\frac{m\pi}{b/2}y) & (m=1\cdots\infty)\\ \sqrt{\frac{1}{b/2}} & (m=0)\end{cases} \tag{3.1.15.d}$$

If the heat input and the travel speed are kept constant at Q and v, then

$$T(w,y,z,t)-T_0=$$

$$\frac{1}{4\sqrt{\pi k}}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\sum_{\substack{n=1\\m=0}}^{\infty}\int_0^t\frac{1}{\sqrt{t-\tau}}\exp\left[-\frac{w^2}{4k}\frac{1}{t-\tau}-(\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b/2})^2)(t-\tau)\right]d\tau$$

$$\cdot\Phi_n(z)\Phi_n(0)\Psi_m(y)\Psi_m(0)$$

$$=\frac{1}{2\sqrt{\pi k}}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\sum_{\substack{n=1\\m=0}}^{\infty}\int_{\frac{1}{\sqrt{t}}}^{\infty}\frac{1}{s^2}\exp\left[-\frac{w^2}{4k}s^2-(\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b/2})^2)\frac{1}{s^2}\right]ds$$

$$\cdot\Phi_n(z)\Phi_n(0)\Psi_m(y)\Psi_m(0)$$

(3.1.16)

Steady-state solution

$$T(w,y,z,\infty)=$$

$$\frac{1}{2\sqrt{\pi k}}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\sum_{\substack{n=1\\m=0}}^{\infty}\int_0^{\infty}\frac{1}{s^2}\exp\left[-\frac{w^2}{4k}s^2-(\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b/2})^2)\frac{1}{s^2}\right]ds$$

$$\cdot\Phi_n(z)\Phi_n(0)\Psi_m(y)\Psi_m(0)$$

$$=\frac{1}{4\sqrt{k}}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\sum_{\substack{n=1\\m=0}}^{\infty}\frac{\exp\left[-\frac{w}{\sqrt{k}}\sqrt{\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b/2})^2}\right]}{\sqrt{\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b/2})^2}} \tag{3.1.17}$$

$$\cdot\Phi_n(z)\Phi_n(0)\Psi_m(y)\Psi_m(0)$$

(6) Model 5

Conditions

(a) Finite thickness and width, and infinite length

(b) Gaussian heat source

(c) Heat flux is described by (3.1.13)

Transient-state model

$$T(w,y,z,t) =$$

$$\frac{1}{4\left(\sqrt{\pi}\right)^3\sqrt{k}}\frac{1}{\sigma^2}\sum_{\substack{n=1\\m=0}}^{\infty}\int_{-\infty}^{\infty}\exp(-\frac{\theta^2}{2\sigma^2})\psi_m(\theta)d\theta$$

$$\cdot\int_{-\infty}^{\infty}d\zeta\exp(-\frac{(w-\zeta)v}{2k}-\frac{\zeta^2}{2\sigma^2})\int_0^t d\tau\frac{1}{\sqrt{t-\tau}}\frac{q(\tau)}{c\rho}$$

$$\cdot\exp\left[-\frac{(w-\zeta)^2}{4k}\frac{1}{t-\tau}-(\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b/2})^2)(t-\tau)\right]\Phi_n(z)\Phi_n(0)\Psi_m(y) \qquad (3.1.18)$$

(note) Velocity v is assumed to be constant.

Steady-state solution

$$T(w,y,z,\infty)-T_0 =$$

$$\frac{1}{8\pi\sqrt{k}}\frac{Q}{c\rho\sigma^2}\sum_{\substack{n=1\\m=0}}^{\infty}\frac{1}{\sqrt{\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b/2})^2)}}\int_{-\infty}^{\infty}\exp(-\frac{\theta^2}{2\sigma^2})\psi_m(\theta)d\theta$$

$$\cdot\int_{-\infty}^{\infty}d\zeta\exp(-\frac{(w-\zeta)v}{2k}-\frac{\zeta^2}{2\sigma^2}-\frac{|w-\zeta|}{\sqrt{k}}\sqrt{\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b/2})^2)})$$

$$\cdot\Phi_n(z)\Phi_n(0)\Psi_m(y) \qquad (3.1.19)$$

Table 3.1 shows the models that we have discussed.

Table 3.1 Conditions given to models

Model	Transient/ steady-state	Thickness	Width	Length	Heat source	Heat flux from surface
Rosenthal's model	Steady-state	Infinite	Infinite	Infinite	Point heat	No
Model 1	Transient	Infinite	Infinite	Infinite	Point heat	Yes
Model 2	Transient Steady-state	Finite	Infinite	Infinite	Point heat	Yes
Model 3	Transient Steady-state	Finite	Infinite	Infinite	Gaussian	Yes
Model 4	Transient Steady-state	Finite	Finite	Infinite	Point heat	Yes
Model 5	Transient (Constant velocity) Steady-state	Finite	Finite	Infinite	Gaussian	Yes

3.2 Choice of model

This section presents how the factors such as thickness, width, heat transfer coefficient, and heat distribution affect the temperature distribution. The purpose of this analysis is to select the model suitable for analyzing the cooling rate and designing the controller. If there are some models that describe well the temperature during welding, the simpler is better. Obviously, the simplest form is the model given by the equation (3.1.1), which was derived by Rosenthal. In this model the factors of the thickness, the width, Gaussian heat source and the heat flux from the surface are not considered. We would like to know whether they are truly negligible.

3.2.1 Effect of thickness

Figure 3.2.1 shows the effect of thickness on the temperature distribution. The steady-state solution in Model 2 is used for this simulation. From this figure we can see that the thickness affects the temperature distribution. When the thickness is less than 20 mm, the temperature distribution is very sensitive to the change of the thickness. Suppose the thickness abruptly changes during welding. The behavior of the cooling rate would then have a big variation by the change. Therefore, we can not ignore the effect of the thickness. We also see that the infinite space model presented by Rosenthal does not necessarily describe the temperature distribution well.

3.2.2 Effect of width

Figure 3.2.2 shows the effect of the width of the workpiece. This result is obtained by the steady-state model in Model 4. If the width is less than 30 mm, the temperature distribution shows a big difference by the change of the width. However, when the width is greater than this value, the temperature distribution is not affected by the change. If we are concerned with the workpiece whose width is bigger than 30 mm, we don't care about the width. We can use the model which assumes the width is infinite. We also note the experiments shown in Chapter 2 were done for the workpieces of 60 mm in width. Therefore, the experiments meet with this condition.

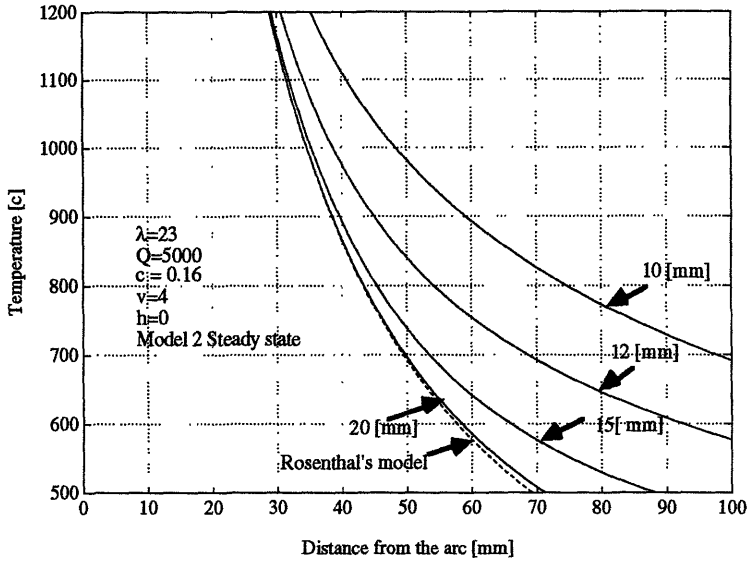


Fig. 3.2.1 Effect of thickness

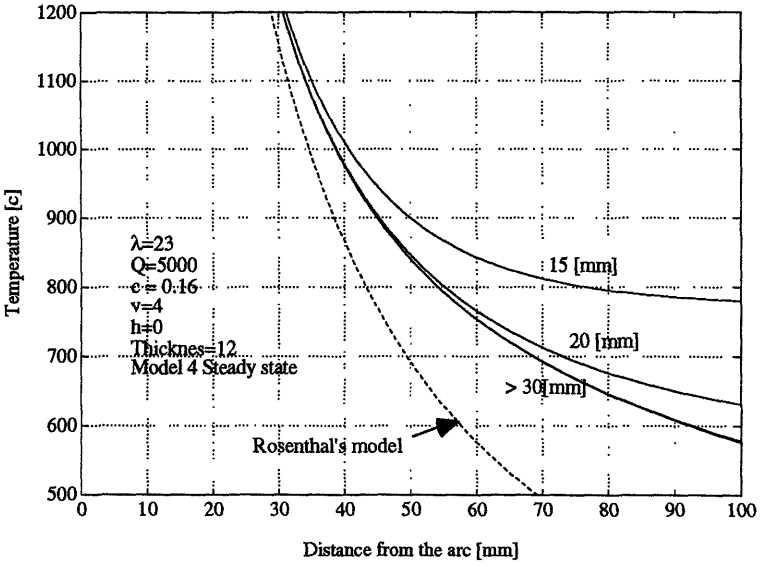


Fig. 3.2.2 Effect of width

3.2.3 Effect of Gaussian distributed heat input

Figure 3.2.3 shows the change of temperature distributions when we change σ in Gaussian heat source. Model 3 is used for this simulation. It seems that the temperature distribution just shifts toward the arc when σ increases. Suppose we attempt to identify the parameters in a model by using the measured temperatures. Then the position of a

temperature plays a key role for the identification. Therefore, we could not ignore the effect of σ, even if it has a small effect on the shape of the temperature distribution.

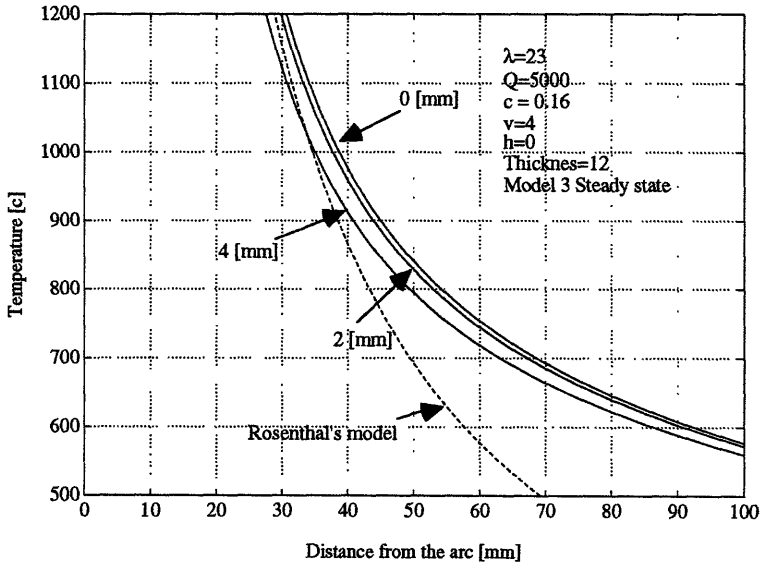


Fig. 3.2.3 Effect of Gaussian heat source

3.2.4 Effect of heat transfer coefficient

Figure 3.2.4 shows the effect of heat transfer coefficient on the temperature distribution. The steady-state solution in Model 2 is used for this simulation. As we can see in the figure, the heat transfer coefficient has an effect on the temperature distribution. Although a big effect is not observed, we can not assume the heat transfer coefficient zero by the same reason for the Gaussian heat distribution. Consider the radiation law, which is written as the fourth power of the surface temperature. There must, therefore, be rather big radiation from the weldment surface.

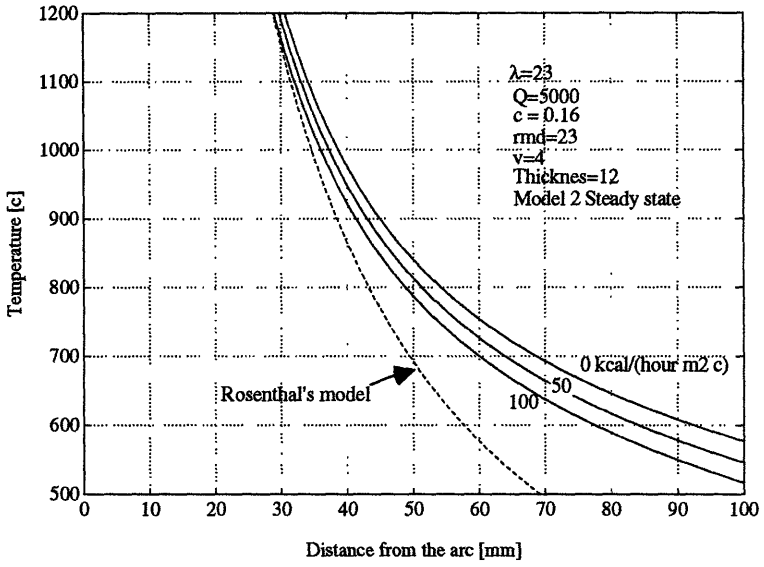


Fig. 3.2.4 The effect of heat transfer coefficient

3.2.5 Choice of model

Let us summarize the results we have obtained.

1. In case that the width is greater than 30 mm, or the welding of two workpieces whose width is greater than 15 mm, we can assume that the width is infinite. Therefore, in the usual case we can ignore the effect of the finite width.
2. The effect of distributed heat source on the temperature distribution would be small. However, it has an effect on the location of the temperature. The location of temperature is important information for identifying the parameters of a model. Therefore, we could not ignore the effect of distributed heat source. By the same reason, we can not ignore the effect of heat flux from the surface.
3. If the thickness is less than 20 mm, the behavior of the temperature distribution varies by the change of the thickness. Since we are concerned with the material with the thickness less than 15 mm, we cannot ignore the effect of the thickness.

From the results mentioned above, Model 3, the model derived under the assumptions: (1) Finite thickness, (2) Infinite width and length, (3) Heat flux from the surface and (4) Gaussian distributed heat source, will be used as the model to analyze and design the control for the cooling rate.

3.3. Dynamic behavior of temperature distribution

We discuss how the selected analytical model given by (3.1.11) behaves for the inputs of travel speed and wire feed rate by simulation. We use the same conditions as those used for the actual measurement in Chapter 2 for the simulation.

3.3.1 Parameters of model

First we have to determine the values of parameters, such as heat transfer coefficient, thermal diffusivity, the distribution of heat source σ and heat input q corresponding to wire feed rate. These values are determined from the experimental results discussed in Chapter 2. The determination is basically based on trial and error. Various simulations are executed, and then the most feasible values of the parameters are selected. The obtained values are as follows.

a. Distribution of heat source σ: 5.0 mm
b. Heat transfer coefficient: 125 kcal/(hour m^2 °C)
c. Thermal diffusivity: 0.035 m^2/h
d. Heat input/Thermal conductivity (q/λ):
162 m°C for wire feed rate of 18 cm/sec
178 m°C for wire feed rate of 25 cm/sec
e. Ambient temperature 30°C

Equation (3.1.10) shows we can not independently identify the heat input and the thermal conductivity (or specific heat with respect to unit volume). we can simply identify the value for the heat input divided by the thermal conductivity (or specific heat with respect to unit volume). As mentioned below it was impossible to select the values which completely describe the experimental results. It must be noted that in analytical models the following assumptions are taken.

1. The parameters have constant values. It is well known that the physical parameters such as thermal diffusivity depend on the temperature. Therefore, the actual parameters vary by the location and time. However, in the model they are assumed constant.
2. Distribution of heat source is not affected by the wire feed rate and the travel speed, and has a constant value.

3.3.2 Simulation

Equation (3.1.10) takes a slightly complicated form. Here we discuss how we can calculate temperature distribution efficiently.
For the centerline surface temperature, equation (3.1.1) has the form:

$$T(w,0,0,t)=\int_0^t q(\tau)\Theta(w+\int_\tau^t v(\eta)d\eta,t-\tau)d\tau \tag{3.3.1}$$

where

$$\Theta(w',t)=\frac{1}{\pi}\sum_{n=1}^{\infty}\frac{1}{4k(t-\tau)+2\sigma^2}\frac{1}{c\rho}\exp\left[-\frac{w'^2+y^2}{4k(t-\tau)+2\sigma^2}-k\alpha_n{}^2(t-\tau)\right]\Phi_n(0)\Phi_n(0) \tag{3.3.2}$$

It can be seen that $\Theta(w',t)$ does not include any terms associated with the inputs. We can also see the term $\int_\tau^t v(\eta)d\eta$ simply shows the distance between the location of the torch at the time t and the past location at the time τ. We also note $\Theta(w',t)$ implies the temperature distribution when an impulse heat input is given to the material, because $q(\tau)=\delta(\tau)$ and $v(\eta)=0$ then

$$T(w,0,0,t)=\int_0^t q(\tau)\Theta(w+\int_\tau^t v(\eta)d\eta,t-\tau)d\tau=\Theta(w,t) \tag{3.3.3}$$

To simplify the integral in (3.3.1) we can use the approximation:

$$T(w,0,0,t)\approx\sum_m q(t-m\Delta\tau)\Theta(w+l(m\Delta\tau,t),m\Delta\tau)\Delta\tau \tag{3.3.4}$$

by taking sufficiently small $\Delta\tau$, where $l(\tau',t)=\int_{\tau'}^t v(\eta)d\eta$

Therefore, once we compute the values of $\Theta(w',\tau)$ for various locations w' and time τ, and store them as a reference table, we can calculate the temperature distribution as a function of time by a simple summation shown in (3.3.4) with the history of the locations of the torch and the heat input.

Figure 3.3.1 shows the shape of $\Theta(w',\tau)$, where $\tau=3\cdot n \quad (n=1,2,\cdots)$. When the temperature distribution during welding is calculated, $\Theta(w',\tau)$ is shifted as shown in

Fig. 3.3.2 in accordance with the location of the torch. Then each set of $\Theta(w', \tau)$ data is multiplied by the corresponding heat input, and the summation of all data gives us the temperature distribution.

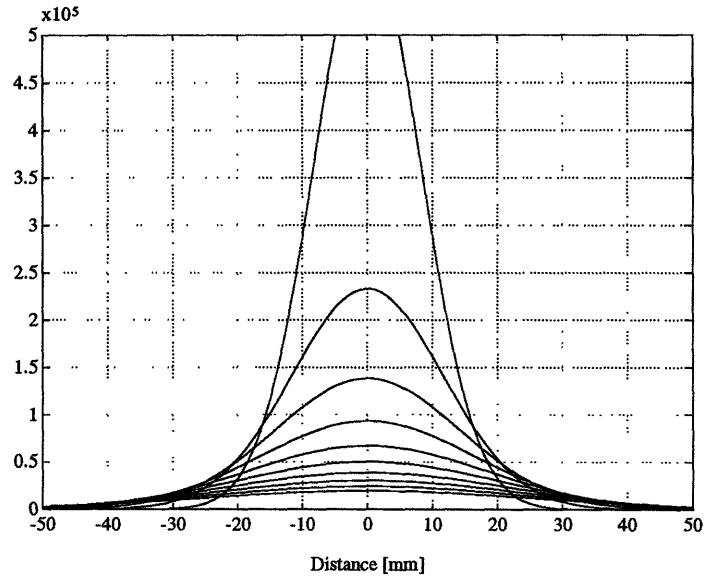


Fig. 3.3.1 Shape of $\Theta(w', \tau)$ data

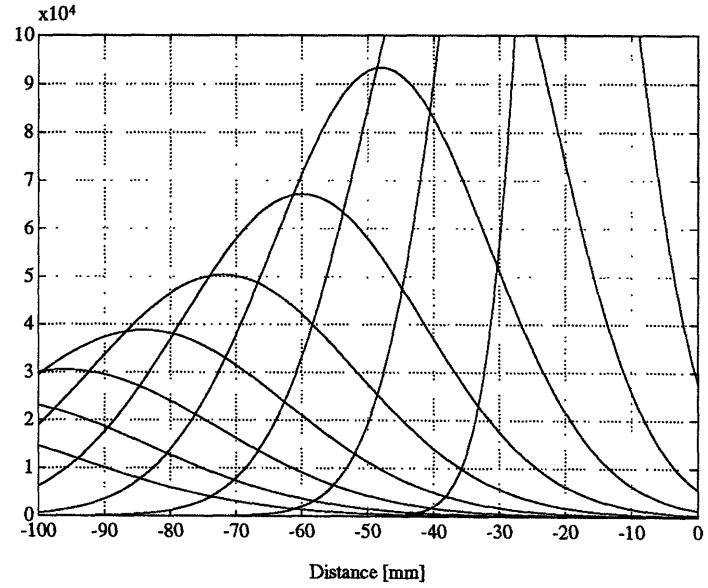
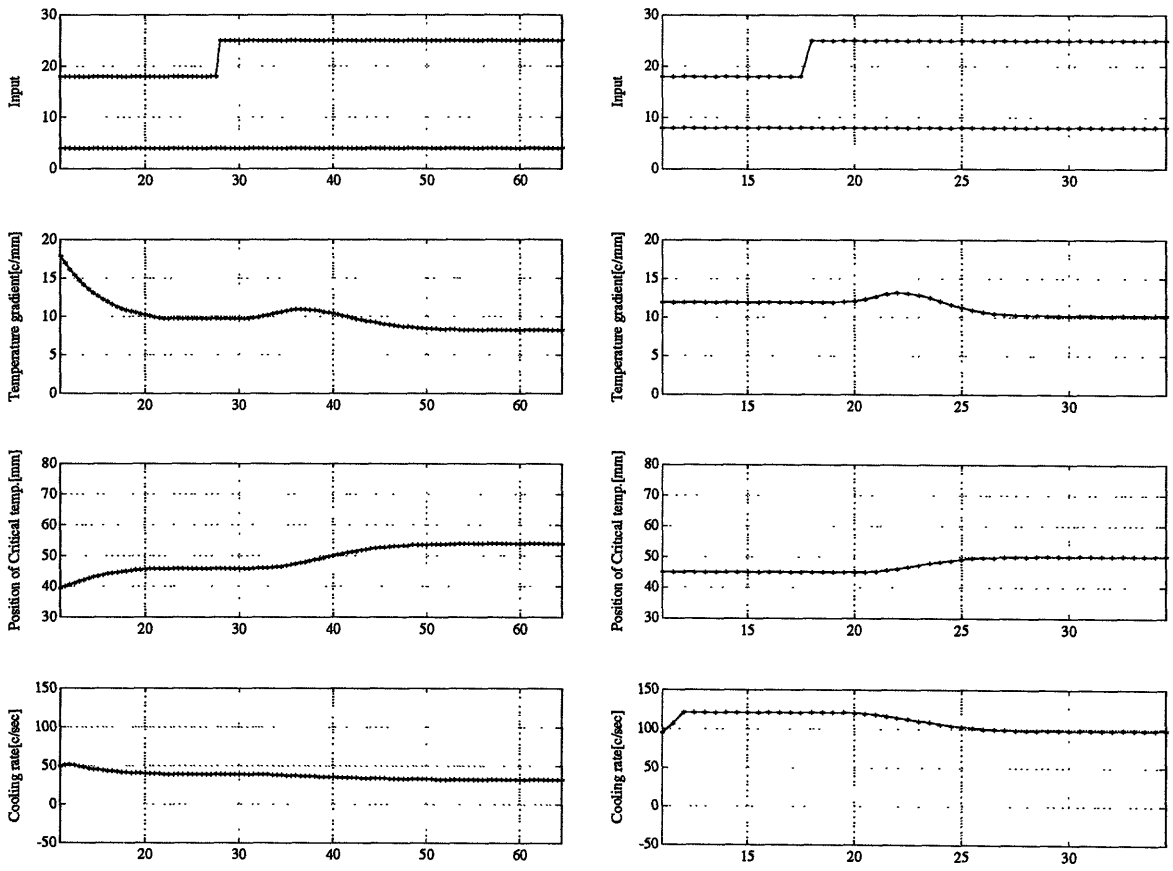


Fig. 3.3.2 Shift of $\Theta(w', \tau)$ data

3.3.3 Dynamic behavior for step-wise change of inputs

(1) Step-wise change of wire feed rate

Figure 3.3.3 shows the simulation results for the response of temperature gradient, location of critical temperature, and cooling rate when we change the heat input in term of q/λ from 162 m°C to 178 m°C, which correspond to the change of the wire feed rate from 18 mm/sec to 25 mm/sec, under the condition same as used in the actual measurement discussed in Chapter 2.



(a) 4 mm/sec in travel speed (b) 8 mm/sec in travel speed

Fig. 3.3.3 Response by step-wise change of wire feed rate

These figures show the following similarity to the actual measurement.

1. The temperature gradient gradually goes up after the step-wise change of the wire feed rate. After it reaches the peak, it gradually goes down.
2. The location of critical temperatures gradually moves away from the arc after the step input.

However, we can also observe the following difference from the actual measurement.

3. Unlike the actual measurement, the cooling rate is slightly affected by the wire feed rate (heat input). In particular, the effect is bigger when the travel speed is 8 mm/sec.

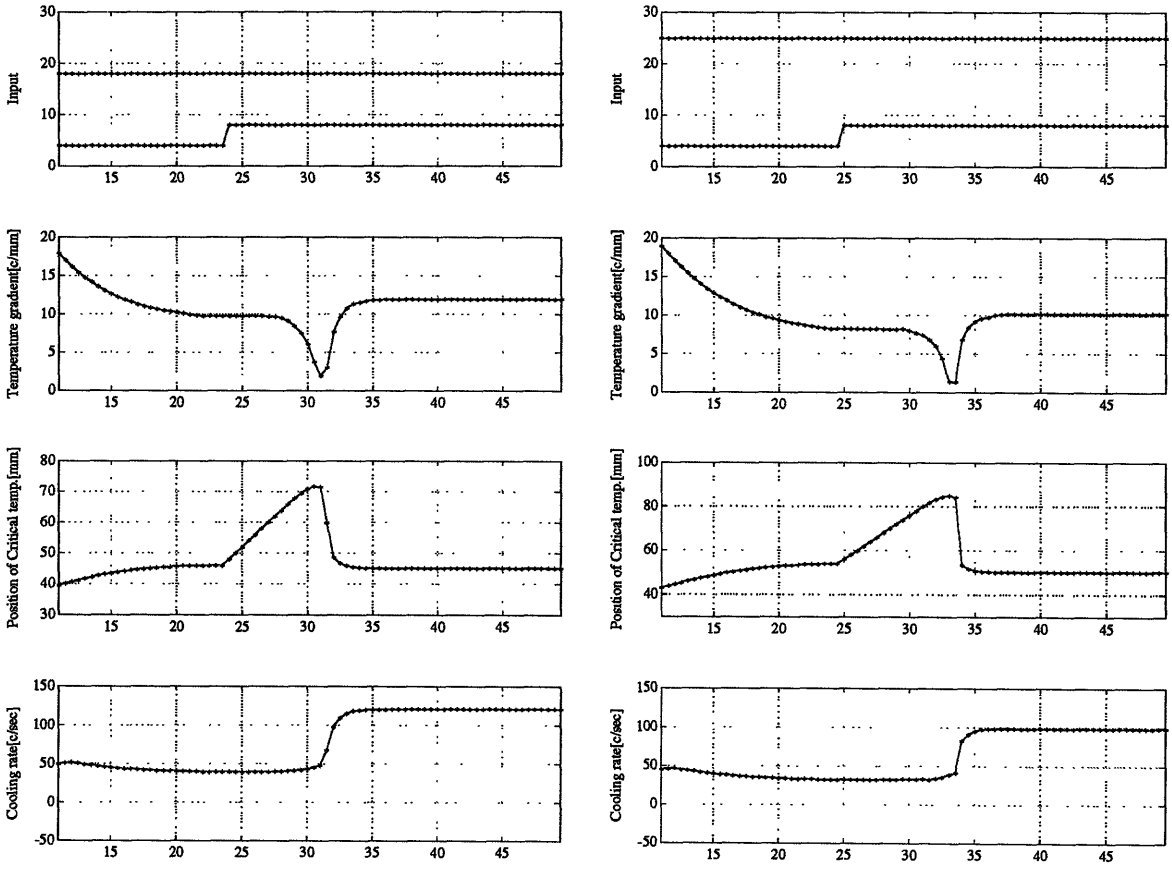
(2) Step-wise change of travel speed

Figure 3.3.4 shows the transient response when the travel speed is changed step-wisely from 4 mm/sec to 8 mm/sec. These figures also show the following similarity to the actual measurement discussed in Chapter 2.

a. The temperature gradient shows a sudden drop and returning.
b. The position of critical temperature gradually moves away from the arc after the step input, and shows an abrupt drop.
c. The cooling rate involves a long time-delay and a first order response

(3) Transient behavior of temperature distribution

Figure 3.3.5 shows the transient change of temperature distribution after the stepwise change of the travel speed. Figure 3.3.6 shows the case of the stepwise change of the wire feed rate. In both figures the actual measurements are also added. Similar responses between the simulations and the experiments are observed in both figures.



(a) 18 cm/sec in wire feed rate (b) 25 cm/sec in wire feed rate

Fig. 3.3.4 Response by step-wise change of travel speed

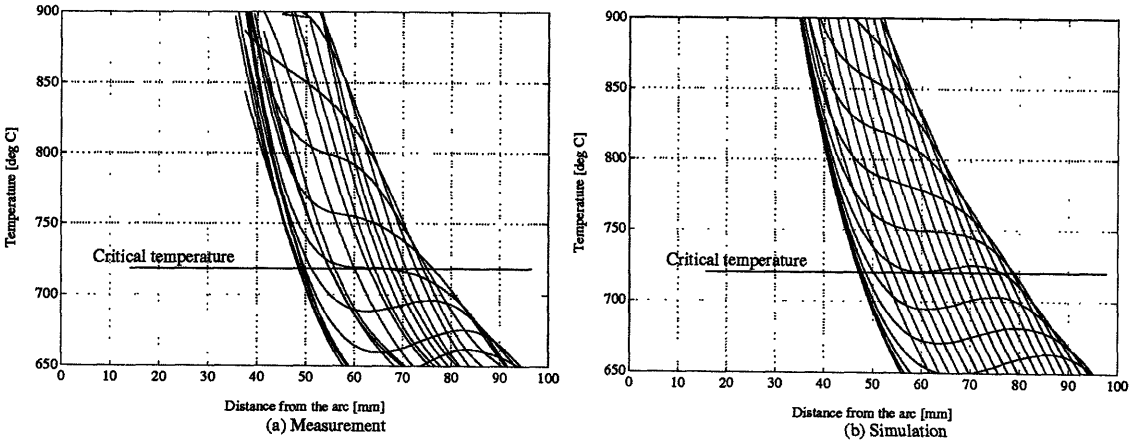


Fig. 3.3.5 Change of temperature distribution after the step-wise change of travel speed

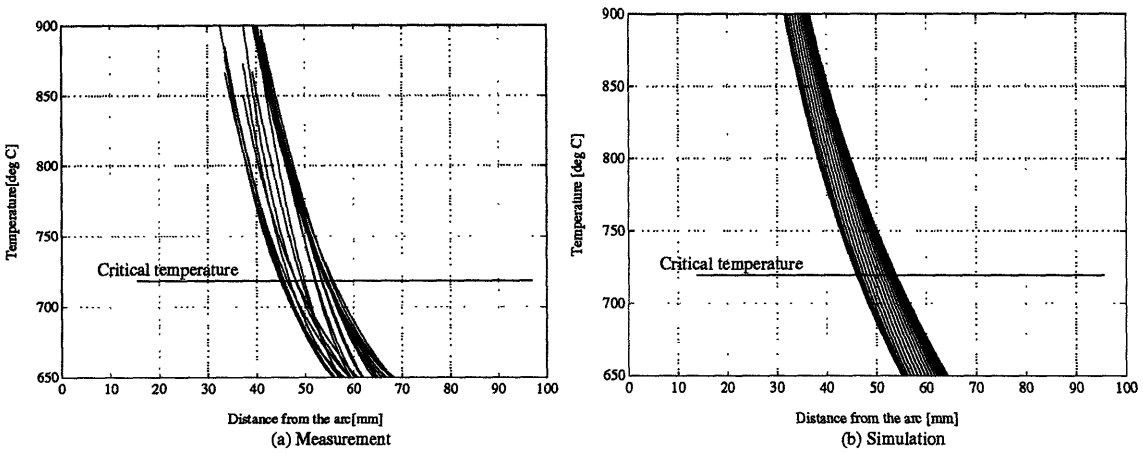


Fig. 3.3.6 Change of temperature distribution after the step-wise change of wire feed rate

3.4 Remarks

1. The simulation by the analytical model shows similar behavior to the actual measurement. Therefore we can say that the measurement system works well for determining how the temperature changes during welding. The equation (3.1.10) could also be used as the model to describe the behavior.

2. As discussed in Section 2.3, the experiment shows the steady-state temperature distribution can be described as an exponential function. These experimental results should permit a simple explanation, however, any interpretation using three-dimensional space models would not succeed. The analytical models say

(a) If the thickness is infinite, the steady-state temperature is written as $T - T_0 \propto 1/w$ as shown in (3.1.1).

(b) If the thickness is finite, and the distance from the arc is somewhat greater than the thickness, the steady-state temperature distribution could be written as $T - T_0 \propto 1/\sqrt{w}$, since in (3.1.8) the modified Bessel function of the second kind and zero order is approximated as:

$$K_0(z) = \sqrt{\frac{\pi}{2z}} \exp(-z)$$

and higher order terms are almost negligible. In our experiment, the condition in **(b)** is satisfied for the location of the critical temperature.

Even if we add the effect of heat transfer coefficient and Gaussian distributed heat input, the tendency above does not change. Therefore, no analytical models would have a satisfactory agreement with the experiment results.

3. The differences between the analytical model and the measurement would come from the following assumptions given to the analytical models.

a. The physical parameters are assumed to be constant. However, the actual parameters vary their values, depending on the temperature. Temperature distribution involves a steep change along the weldment. Therefore, the actual parameters also have a big variation along the weldment. The variation is not considered in the analytical model.

b. Phenomena in the weld pool, such as convection, are not taken into consideration. It is reported that the heat transfer in the weld pool is completely different from that by the heat conduction (Heiple et al. (1990), and Oreper et al. (1983)). Our interest is in the region after the solidification. However, the temperature in the solidified region must be affected by that in the weld pool.

c. In GMAW mass transfer by wire feed takes place simultaneously with the heat supply. However, it is not considered in the analytical model.

It is quite hard to establish a model to relax these assumptions, even if we resort to numerical methods. However, we could not say clearly the numerical models give us more insight than the analytical model. A model will be used for designing the control for the cooling rate. From this point of view, the derived analytical model would be enough for this purpose.

Chapter 4

Problems in Control of Cooling Rate

This chapter looks at the problems for the control of centerline cooling rate. As the experiments in Chapter 2 suggest, the cooling rate should be controlled by travel speed. Therefore the control by travel speed will be considered. The purpose of this chapter is first to observe the nature of the cooling rate dynamics, and second to see how conventional PI control performs for the process.

4.1 Nonlinearity and time-delay

Figure 4.1.1 shows the step-response of the cooling rate when the travel speed is changed. The response is obtained by simulation discussed in Chapter 3. The same condition as in Chapter 3 is also used for the simulation. It is easily seen that the response involves a time-delay and a nonlinearity. (See the responses when we change the travel speed from 6 mm/sec to 8 mm/sec, and 6 mm/sec to 8 mm/sec. The time-delay and the time constant differ in both cases.) The nonlinearity and time-delay are characterized as follows.

1. The time-delay varies from 2 sec to 6 sec, depending on the value of the travel speed and the initial state. In particular, the initial state seems to have a strong effect on the time-delay.
2. The cooling rate takes a first-order or second-order response after the time-delay. If it takes a second order response, it is an overdamped response, since it also does not show any overshoot and undershoot behavior. When we increase the travel speed, it has a tendency to show a second order response, while it shows a first order response when the travel speed decreases. The parameters characterizing the response such as time constant changes by the input and the state of the system.
3. The steady-state behavior also shows a nonlinearity. The steady-state change of the cooling rate is not proportional to the change of the input, the travel speed.

The characteristics above are quite troublesome when we design the control, because any control design method based on linear systems can not deal with such problems. It is sometimes suggested to linearize a nonlinear system. However, control based on the linearized system would be easily degraded by the nonlinearity and the time-delay. Therefore it is important to design the control to accommodate the time-delay and the nonlinearity of the system.

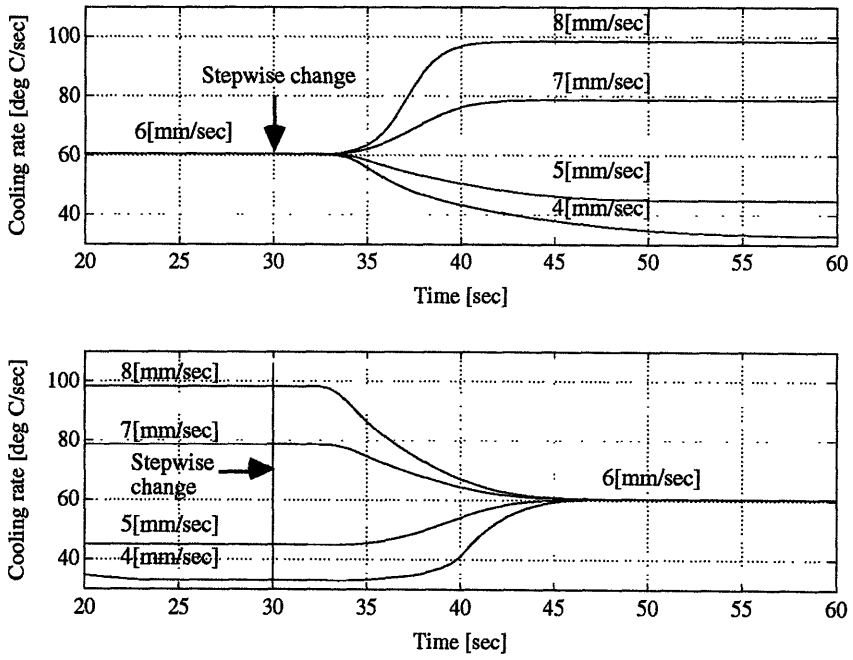


Fig. 4.1.1 Step response by travel speed

We have discussed the difficulty for designing the control, which comes from the nonlinearity of the system. However, the process is not so highly nonlinear that linear control does not work. As shown below, conventional PI control works for controlling the cooling rate, although the performance is not so good. When we increase the travel speed, the cooling rate always increases. The decrease of the travel speed necessarily decreases the cooling rate. This simple but essential relation between input and output ensures the applicability of PI control.

4.2 Results of PI control

PI control is widely used in process control. The implementation is quite easy. We simply determine the tuning parameters by an empirical method or by trial and error. It is applicable to almost all plants, and the above fact still attracts many engineers, although the idea of PI control was born more than one century ago.

However, the use of PI control for higher order or nonlinear systems sacrifices the performance. The objective of this section is to show how poor the PI control is for cooling rate. All discussions are based on the simulations using the analytical model that is derived in Chapter 3.

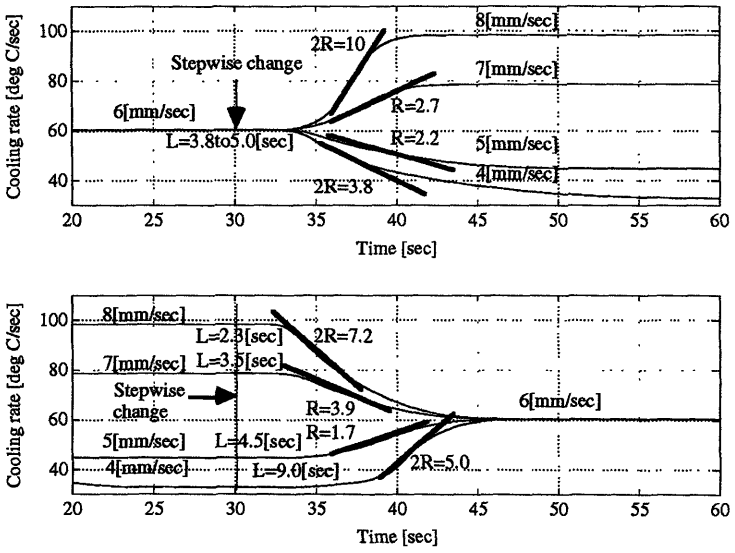
4.2.1 PI controller design

Let us design PI controller by the empirical Ziegler and Nichols method. (See Ogata (1990)) The rough observation of the figure shows

(a) The value of R varies from 1.7 to 5.

(b) The value of L varies from 2.3 to 9.0 [sec].

where R is the steepest slope of the response for unit step input. L denotes a time-delay when the delay-free part of the response is assumed to be of first order as shown in Fig. 4.2.1.



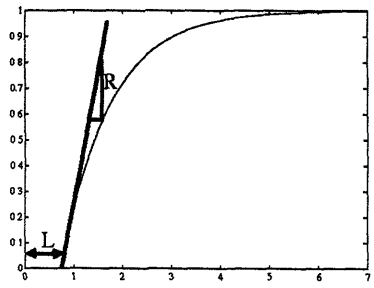


Fig. 4.2.2 Step response of a first order system

Fig. 4.2.1 Values for R and L in the response of cooling rate

Using Ziegler and Nichols method, the constants in PI controller $K(1+1/T_I s)$ are given by

$$K = 0.9/RL = 0.9/5.9 = 0.02$$
$$T_I = L/0.3 = 9/0.3 = 30 \tag{4.2.1}$$

taking into consideration the worst case of the approximation by a first order system. The PI controller above is written in the pulse transfer function as

$$u(z) = K(1+\frac{T}{T_I(1-z^{-1})})e(z) = \frac{(K+\frac{K}{T_I})-Kz^{-1}}{(1-z^{-1})}e(z) \tag{4.2.2}$$

where $e(t)$ is output error, $u(t)$ is input to the plant and T is the sampling period. Therefore, the control is given by

$$\begin{aligned} u(t) &= u(t-1)+K(e(t)-e(t-1))+\frac{KT}{T_I}e(t) \\ &= u(t-1)+0.02(e(t)-e(t-1))+0.02\cdot 0.0167e(t) \end{aligned} \tag{4.2.3}$$

4.2.2 Simulation results

Figure 4.2.3 shows the response by simulation when the PI control with the above constants is used. In the simulation, we use the analytical model discussed in Chapter 3, assuming the material is same as those used in experiments in Chapter 2. We also take 25 cm/sec as the wire feed rate, and the value of the travel speed is bounded within the region from 4 mm/sec to 8 mm/sec. The cooling rate shows an oscillation in the response. Therefore, Ziegler and Nichols method does not succeed in determining the constants of PI controller.

Since there is no other empirical methods than Ziegler and Nichols method for designing the PI controller, we will determine the constants of the controller by trial and error. Figure 4.2.4 shows the responses when we change the gain K, fixing the value of Ti 30. When K is 0.0015, we obtain the best response.

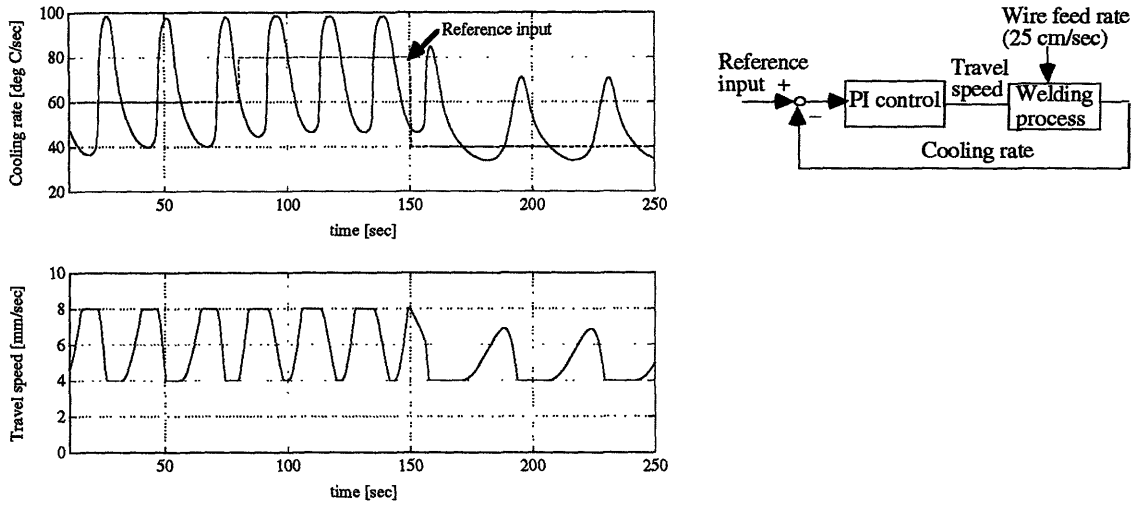
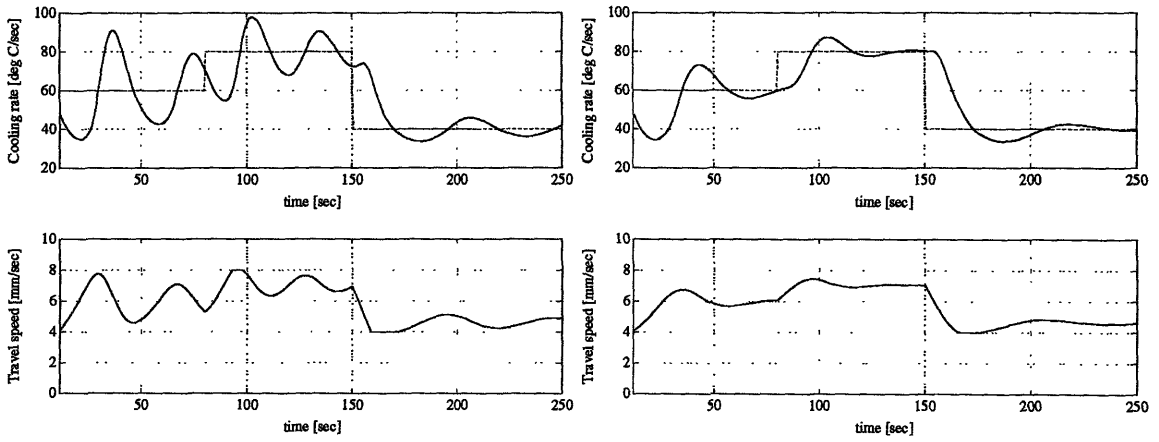
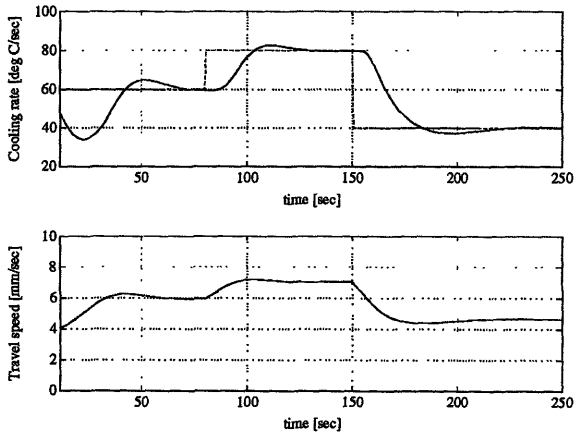


Fig. 4.2.3 Response of PI control (Ziegler and Nichols method)

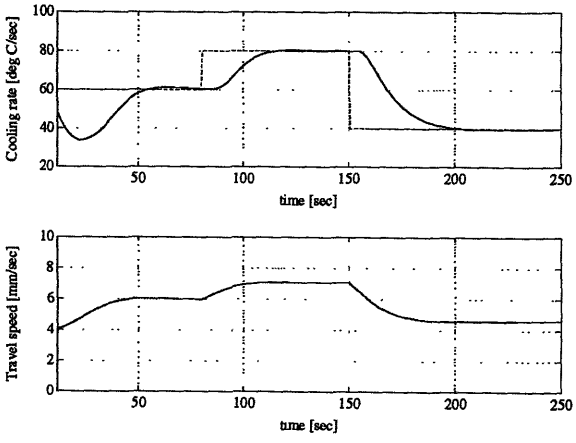


(1) K=0.005

(2) K=0.003



(3) K=0.002



(4) K=0.0015

Fig. 4.2.4 Response of PI control

4.2.3 Performance of PI control

Figure 4.2.5 shows the step response of PI control for some reference signals. The PI controller takes the parameters determined in Section 4.2.2. The figure shows the following performance of this control.

1. Due to the presence of a nonlinearity and a varying time-delay, the settling time depends on the reference input.
2. The settling time is from 30 sec to 50 sec.

The PI control works for our problems. However, the performance is not so satisfactory, showing the slow response due to the presence of the nonlinearity and the time-delay.

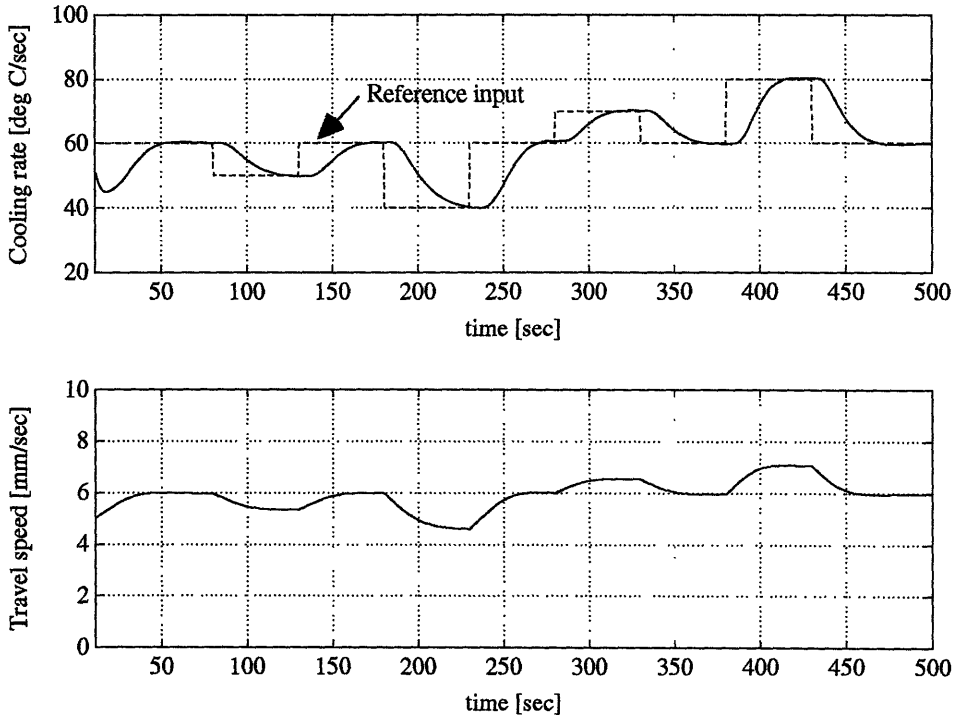


Fig. 4.2.5 Responses for stepwise reference signals

4.3 Conclusions

The cooling rate dynamics by travel speed shows a nonlinearity and a varying time-delay. As for the control of the cooling rate, PI control works. However, the performance is not satisfactory due to the presence of the nonlinearity and the time-delay. In order to get better performance, we have to design a control that accommodates a nonlinearity and a varying time-delay.

Chapter 5

Control for Nonlinear and Time-Delay Systems

In Chapter 4 we discussed the importance of control that copes with a nonlinearity in the dynamic behavior of the cooling rate. The controller should also accommodate a time-delay that varies by the input and the state of the system. In what follows, we would like to discuss how we can deal with such problems. Several methods have been presented by many researchers to tackle nonlinear control problems. However, they have some limitations. When we design a control system for the cooling rate, we must, therefore, understand their limitations, and decide which methods are suitable.

Section 5.1 discusses problems in the design of a control system. Section 5.2 reviews linear and nonlinear models for discrete-time systems. Section 5.3 presents neural networks, which attract much interest as a tool of nonlinear control. Section 5.4 deals with linear and nonlinear identification methods. Section 5.5 describes controls which deal with a time-delay. Nonlinear control methods and adaptive controls are treated in Section 5.6 and Section 5.7 respectively. Finally a feasible control system for the cooling rate is discussed in Section 5.8, based on the discussions in Section 5.1 through Section 5.7.

5.1 General review of control system design

The design process of a control system involves many steps. A typical textbook on control theory says the scenario for designing a control system can be described as follows (Doyle et al. (1992)):

1. Study the system to be controlled and decide which types of sensors and actuators will be used and where they will be placed.
2. Model the resulting system to be controlled.
3. Simplify the model if necessary so that it is tractable.
4. Analyze the resulting model; determine its properties.
5. Decide on performance specifications.
6. Decide on the type of controller used.

7. Design a controller to meet the specs, if possible, if not, modify the specs or generalize the type of controller sought.
8. Simulate the resulting controlled system, either on a computer or in a pilot plant.
9. Choose hardware and software and implement the controller.
10. Tune the controller on-line if necessary.

For the item 1 we discussed the nature of the cooling rate in Chapter 2 through Chapter 4. It was shown that the cooling rate should be controlled by the travel speed. However, the response by the travel speed shows a highly nonlinear dynamics with a varying time-delay. From the item 2, they require our decisions, on which the performance of the controller will strongly depend.

In what follows we shall discuss what we should consider for the design of the controller for the cooling rate. The detailed discussion and review will be found in the later sections.

5.1.1 Models

When we design a controller, we must always consider what kind model should be used. As shown in Chapter 3, we derived the analytical model for the temperature distribution in welding. A control system using this analytical model would be quite difficult to design. Control is a system which determines appropriate inputs from the measured outputs via a control law. In a sense, the control law requires an inversion of the plant model from the output to the input so that we can determine a suitable input that produces a satisfactory output. Unfortunately such manipulation of the analytical model is quite difficult.

Therefore, the model should be simple and available to the design of the controller. Several representations of models have been presented. The discussion and review of them will be held in Section 5.2.

We need to select the most appropriate representation. Obviously the representation of a controller is closely related to that of the selected plant model. Therefore we also have to pay attention to what type of model the control requires.

5.1.2 Objectives of control

In general, there are two types of control objectives. One is a regulation problem, whose objective is to maintain the output at a constant value. The other is a tracking problem, or servo problem, whose objective is to follow the output to a desired trajectory.

In general the objective of the cooling rate control is to maintain the cooling rate at a specified value. Therefore, it belongs to a regulation problem.

5.1.3 Stability and Performance

Stability is the fundamental issue of control systems. If a plant is not stable, the controller should stabilize the plant. If the plant is stable, the controller must not cause any unstable behavior. Obviously the dynamic behavior of the cooling rate is stable since constant heat yields a bounded cooling rate. Therefore we should think of the latter case.

Next what we should consider is how the controlled system performs. The typical system configuration that includes the controller is shown in Fig. 5.1.1. In the figure we have three uncertainties that are not included in the control system:

(1) Unknown plant perturbation

(2) Unknown sensor noise

(3) Unknown disturbance

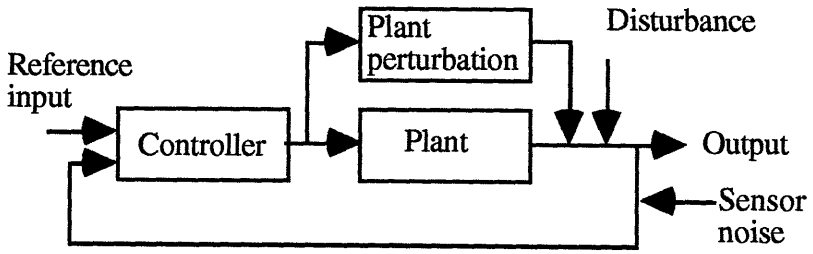


Fig. 5.1.1 Block diagram of control system

We are concerned with the regulation problem of the cooling rate. Therefore the performance of the control system should be given by the statements:

1. The system rejects the disturbance as fast as it can.

2. The system is as insensitive to the noise as it can.

3. (Robust stability) Even if the plant dynamics is perturbed from the (nominal) plant model that is used for the design of the controller, the system must be stable.

4. (Robust performance) Even if the plant dynamics is perturbed from the nominal plant model, we can have similar performance for the disturbance rejection and the insensitivity to noise.

In manufacturing systems, the most troublesome uncertainty is the plant perturbation. Consider the cooling rate control. We can easily see that there are many factors that change the dynamics. For example, the variation in the thickness and the properties of the material yields the change of the heat transfer, which in turn changes the dynamics of the cooling rate.

Also, the measurement of the cooling rate is easily contaminated by noise. (See the actual measurement result in Chapter 2.) Therefor, it is quite important for the system to be insensitive to the noise.

The most influential exogenous input, or the disturbance would be the wire feed rate for the control of the cooling rate by the travel speed. However, the wire feed rate has a small effect on the cooling rate as we discussed in Chapter 2. Therefore, it seems there is no significant disturbance in this system. However, we can easily find the factors that disturb the system. If there is a drift in the welding machine, it would yield an output error.

5.1.4 Control scheme

We will consider the following three types of control schemes which have a feed-back structure.

Feed-back control with fixed parameters

In this scheme the controller has a fixed structure with fixed parameters. The dynamic relation between the inputs of the controller, which are usually connected to the output of the plan, and the outputs, which are linked to the inputs of the plant, is time-invariant. Many methods, such as a classical control based on frequency domain, a state feed-back control and an optimal control, are available for a linear system.

Adaptive control

The dynamic relation between the input and the output of the controller is time-variant. The basic concept of this scheme is that the controller always detects the change of the plant dynamics, interprets it and continuously adjusts the control. The precise discussion of adaptation can be found in the book by Narendra and Annaswamy (1989).

Even if the system is assumed linear and the controller has a linear structure, the adaptive control scheme requires a consideration for the nonlinearity. (Consider the controller based on the linear control law whose parameters also changes by the adaptation law. Obviously such a system is nonlinear.)

Some nonlinear adaptive control schemes using neural networks have been presented. The idea is based on the optimization /minimization of the output error by a nonlinear minimization algorithm.

Iterative learning control

This method is regarded as a feed-forward control, but it iteratively determines the next series of inputs fed into the plant from the past operation results so that the plant gives us desired outputs. This idea is sometimes beneficial in manufacturing processes because of the repeated operations for the products with the same specifications. Extensive discussion can be seen in the book by Moore (1992).

Since iterative learning control scheme takes a feed-forward configuration, it can not cope with uncertainties taking place during operation. This method is suitable for the process that has strong nonlinearity, complexity caused by a large number of parameters, and highly time-variant nature, which make it hard to implement the on-line feed-back loop. The concern of this scheme is to establish a control to achieve the specified output, allowing the error caused by the disturbance during operation. Therefore, if we can implement the on-line feed-back configuration, this scheme is out of interest. As shown in Chapter 4, PI control works for the cooling rate. Therefore, we do not have to resort to iterative learning control scheme. We will constrain our interest on the two schemes, feed-back control with fixed parameters and adaptive control.

5.1.5 What shall we discuss in this chapter?

Figure 5.1.2 shows what we shall discuss in this chapter. The diversity of control theory is quite confusing when we design the control. In this chapter we pick up methods feasible for our problem, and hold a concise discussion about them. In particular, we will pay attention to adaptive control for linear controllers, and neural networks for nonlinear controllers. A neural network is merely a model describing a nonlinear system. Therefore, it is important to see how we can design a controller employing neural networks, and that will be the central issue of our discussion.

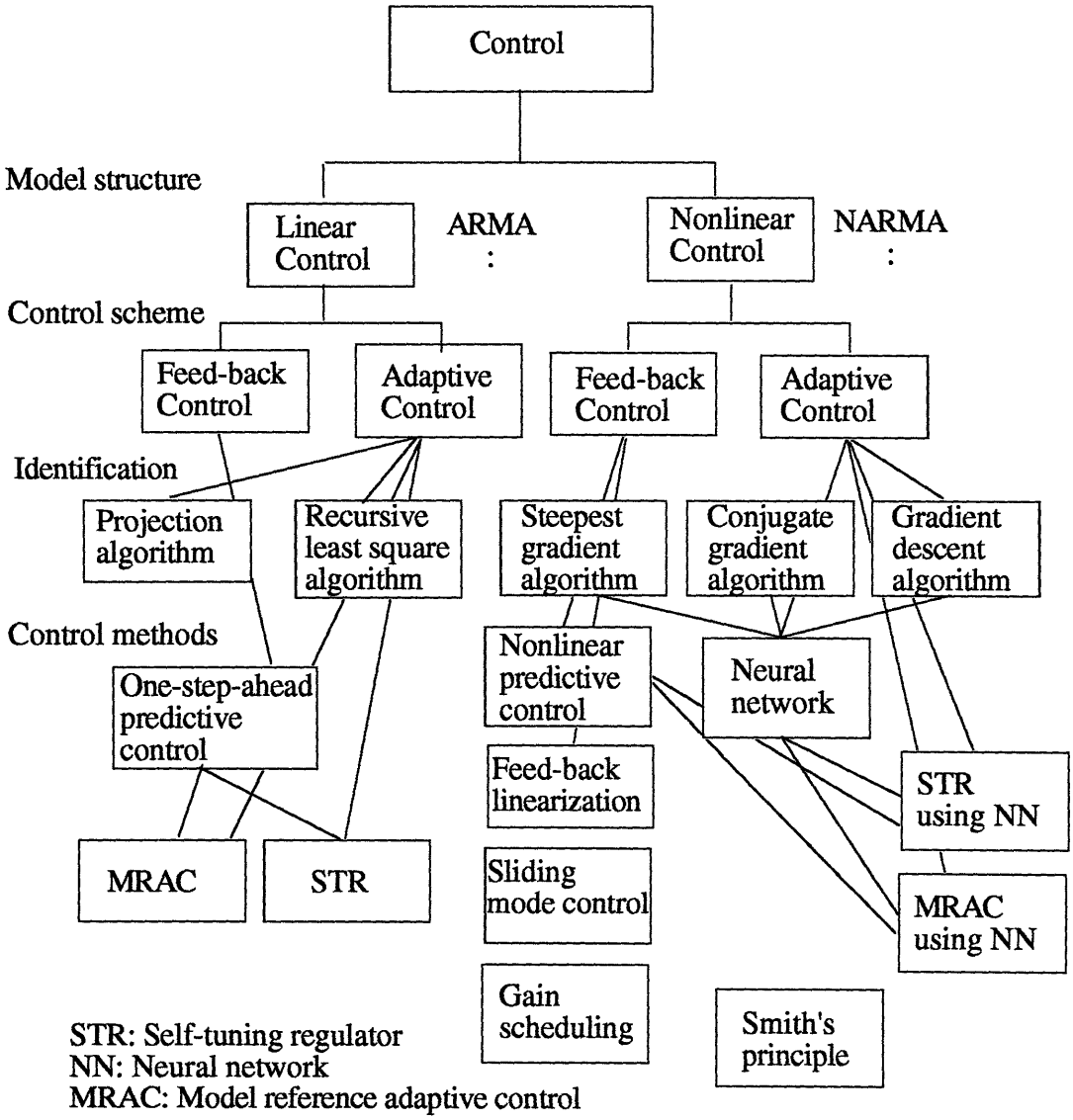


Fig. 5.1.2 Feasible control methods

5.2 Models for discrete-time systems

This section reviews the representations of models which are widely used for control systems. To simplify the discussion, we will pay attention to discrete-time systems. We also constraint our interest to a single-input-single-output (SISO) case, since the system for the cooling rate is SISO.
In Section 5.2.1 we discuss considerations on linear models and nonlinear models. In Section 5.2.2 the representations of linear systems and nonlinear systems are reviewed.

5.2.1 Advantages and disadvantages of linear models and nonlinear models

Linear control theory has been developed for decades with a variety of well-established methods and successful industrial applications. Therefore, it is quite attractive to assume that the system is linear and design the control, using many sophisticated methods. However, the dynamic system which we want to control is nonlinear, and we also have to be concerned with an unknown plant perturbation. Therefore, we have to design the control to be robust against uncertainties caused not only by a nonlinear nature of the system but also by a plant perturbation if we use a linear model.

The other attempt is to use a nonlinear model and design a nonlinear controller. However, the treatment of nonlinear systems is less established than that of linear systems. Limited numbers of methods are available and practical experience is limited. The analysis of nonlinear systems is quite difficult. It is well known that almost all control theories for linear systems do not work for nonlinear systems.

The discussion above shows the characteristics of linear models and nonlinear models.

1. **Linear models:** The analysis is established by many theories. However, when we decide on the control, it should be a system that can hold robustness against uncertainties that comes from the nonlinear process and variations in the process.
2. **Nonlinear models:** The analysis is hard, and the theories concerning it are limited. However, nonlinear controllers might show better performance than linear controllers.

Therefore, we always have to consider the possibility of both models. Even if the system is nonlinear, we can not discard linear models.

5.2.2 Linear models

Well-known three discrete-time representations that are typically used for the control of SISO linear systems are shown.

Discrete state space equation

$$\begin{aligned} &\boldsymbol{x}(t+1) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{b}u(t) \\ &y(t+1) = \boldsymbol{c}^t\boldsymbol{x}(t) + \boldsymbol{d}u(t) \\ &\boldsymbol{x}(t) \in R^n \end{aligned} \tag{5.2.1}$$

where n denotes the order of the system, $\boldsymbol{x}(t)$ is the state at the time t, $u(t)$ is the input variable, $y(t)$ is the output variable, $\boldsymbol{A}$ is the $n \times n$ matrix, and $\boldsymbol{b}$,$\boldsymbol{c}$, and $\boldsymbol{d}$ are the n-vectors.

Pulse transfer function

$$\begin{aligned} &y(z) = z^{-d}\frac{B(z)}{A(z)}u(z) \\ &\text{where} \\ &A(z) = a_0 + a_1 z^{-1} \cdots a_n z^{-n} \\ &B(z) = b_0 + b_1 z^{-1} \cdots b_m z^{-m} \end{aligned} \tag{5.2.2}$$

where $m+d \le n$ and n denotes the order of the system, $y(z)$ and $x(z)$ denote the z-transforms of the output sequence and the input sequence respectively, $z^{-d}B(z)/A(z)$ provides the pulse transfer function of the discrete-time system.

Autoregressive Moving-Average (ARMA) model

$$\begin{aligned} &y(t) = -a_1 y(t-1) \cdots -a_n y(t-n) + b_1 u(t-1) \cdots b_m y(t-m) \\ &\text{It is also written as} \\ &A(q^{-1})y(t) = B(q^{-1})u(t) \end{aligned} \tag{5.2.3}$$

where q^{-1} is the backward shift operator, and then

$$A(q^{-1}) = 1 + a_1 q^{-1} \cdots a_n q^{-n}$$

$$B(q^{-1}) = b_0 + b_1 q^{-1} \cdots b_m q^{-m}$$

For the cooling rate we can only have the dynamic relation between the input (transfer speed) and the output (cooling rate) because we can not precisely define the state by the physical interpretation of the system. Therefore, when we identify the system, the transfer function model and ARMA model are available. For the linear system, it is easy to transform the obtained transfer function to the state space equation, although we may not measure the states.

5.2.3 Nonlinear models

As an extension of the representations of linear systems, the following two types of nonlinear representations can be considered.

State space equation

$$\begin{aligned} \boldsymbol{x}(t+1) &= \boldsymbol{f}(\boldsymbol{x}(t), u(t)) \\ y(t+1) &= g(\boldsymbol{x}(t), u(t)) \\ \boldsymbol{x}(t) &\in R^n \quad n\text{: Dimension of state} \end{aligned} \tag{5.2.4}$$

Nonlinear Autoregressive Moving-Average (NARMA) model

$$y(t) = f(y(t-1), \cdots, y(t-n), u(t-1), \cdots, y(t-m)) \tag{5.2.5}$$

Note that in general the transformation from a NARMA model to the state space equation is quite difficult. For cooling rate control, since we only know the dynamics between the output and the input, only NARMA model can be used.

A nonlinear control theory usually assumes a model is given by one of the above representations. Therefore, if an available control design method is based on state space equations, we can not avoid giving a constraint to the model structure or giving up the method.

Recently many researchers have shown an active interest in nonlinear controls using neural networks, motivated by the fact any nonlinear mapping can be approximated by neural networks. It will be found in Section 5.4 that the neural networks have a suitable configuration describing NARMA. We can also apply neural networks for a model represented by state space equations. For the model, however, we would need a physical interpretation for the states. We will discuss the general configuration of neural networks there.

5.3 Neural Networks

In this section, we discuss how neural networks behave for nonlinear problems. The neural networks have attracted much interest of researchers for a decade. Therefore, quite many literatures have been presented. For the general discussion about neural networks we can refer to the book by John Hertz et al. (1991). As for control systems using neural networks, we can refer to the book edited by White and Sofge (1992). The literature presented by Narendra and Parthasarathy (1990) is quite important from the viewpoint of control. They presented some structures for identification and control employing neural networks.

5.3.1 Multilayer neural networks and back-propagation

The concept of neural networks can be described as:

1. Many simple and identical multi-input-single-output elements are connected to one another, and make a network.
2. If some inputs are given to a part of the network, another part gives us the outputs.
3. Parameters in the elements are adjusted to reduce the error between the outputs of the network and desired values.

Such configuration yields an interesting behavior. They are interpreted as a model of biological neural systems. However, from practical point of view, it is simply regarded as one of methods describing a nonlinear multi-input-multi-output mapping. Neural networks widely used have the following configuration as shown in Fig. 5.3.1.

$$\boldsymbol{y} = NN[\boldsymbol{u}] = \Gamma[\boldsymbol{w}^3\Gamma[\boldsymbol{w}^2\Gamma[\boldsymbol{w}^1\boldsymbol{u}]]] \tag{5.3.1}$$

where $\boldsymbol{w}^i$ is a weight matrix and Γ is a nonlinear operator with identical nonlinear elements γ.

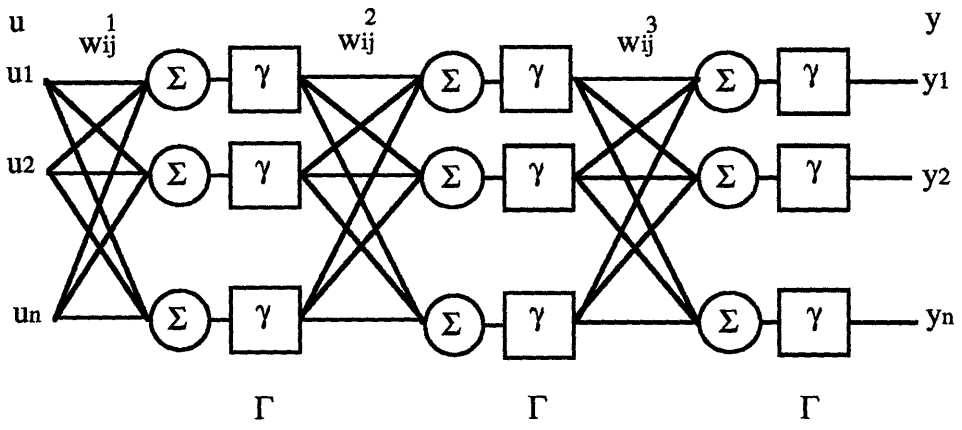


Fig. 5.3.1 Feed-forward multilayer neural network

A logistic function, a sigmoid function and a hyperbolic tangent function are usually used as the nonlinear element (γ). This configuration is sometimes called a multilayer feed-forward network. The most important result for neural networks is that they can approximate any nonlinear function with a hidden layer comprising many nonlinear elements. (It is proved by Funahashi (1989) and other researchers.)

Connectionists, who deal with the computation of a parallel and distribution system, suggest that neural networks should iteratively adjust themselves whenever they detect errors. From this point of view, the parameters should be successively adjusted by the errors, even though this method does not necessarily ensure the good convergence.

One of the most convenient methods of the adjustment is the error back-propagation method, which is based on gradient descent minimization discussed in Section 5.4. It is simply written as: $\boldsymbol{w}^{[k+1]} = \boldsymbol{w}^{[k]} - \eta \cdot \nabla_{\boldsymbol{w}} J$, where k is the number of iteration, $\boldsymbol{w}$ is the weight vector comprising all weights in a neural network, η is a step size, and J is a cost function determined by the error. The chain rule yields the architecture for the update as shown in Fig. 5.3.2. The errors seem to be propagated backward from the final layer to the first layer. This algorithm can be easily implemented. We shall discuss the detailed algorithm for the error back-propagation in Chapter 7

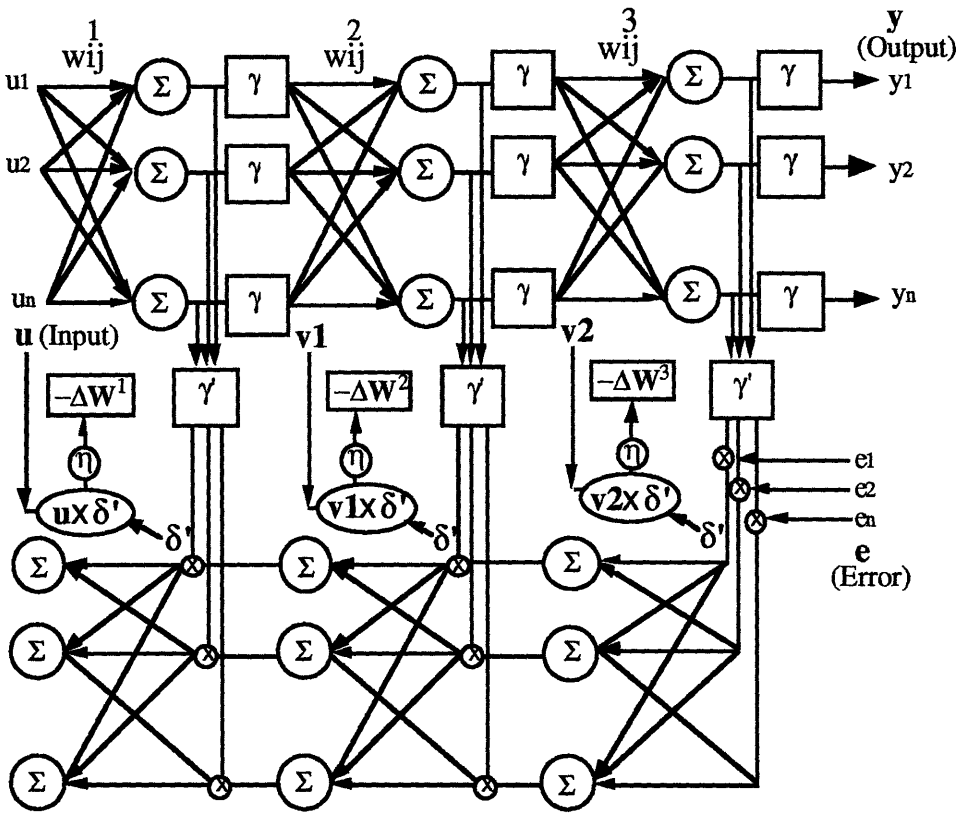


Fig. 5.3.2 Back-propagation

Usually the back-propagation requires quite many iterations. Therefore, we can not always say neural networks are suitable for real-time applications.

Recently many applications using neural networks have been presented. However, these applications have primarily dealt with a static mapping by neural networks. As Hopfield presented, neural networks can have a dynamic property. (See the book by Hertz et al. (1991)) Thus, they expand their capability, used as components in dynamic systems.

5.3.2 Recurrent networks and identification

Hopfield presented neural networks which have a dynamic system configuration, recurrent networks. Narendra et al. (1990) extended his idea and presented the following 4 models using neural networks, which can describe nonlinear dynamic system.

1. Model 1

$$y(t+1) = \sum_{i=0}^{n-1} \alpha_i y(t-i) + NN[u(t), u(t-1), \cdots, u(t-m+1)] \tag{5.3.2}$$

2. Model 2

$$y(t+1) = NN[y(t), y(t-1), \cdots, y(t-n+1)] + \sum_{i=0}^{m-1} \beta_i u(t-i) \tag{5.3.3}$$

3. Model 3

$$\begin{aligned} y(t+1) = {} & NN_1[y(t), y(t-1), \cdots, y(t-n+1)] \\ & + NN_2[u(t), u(t-1), \cdots, u(t-m+1)] \end{aligned} \tag{5.3.4}$$

4. Model 4

$$y(t+1) = NN[y(t), y(t-1), \cdots, y(t-n+1), u(t), u(t-1), \cdots, u(t-m+1)] \tag{5.3.5}$$

We can easily see that these configurations represent NARMA model. Model 4 is the most general model. However, Model 2 is particularly useful for control purposes because the relation between the output $y(t+1)$ and input $u(t)$ is linear.

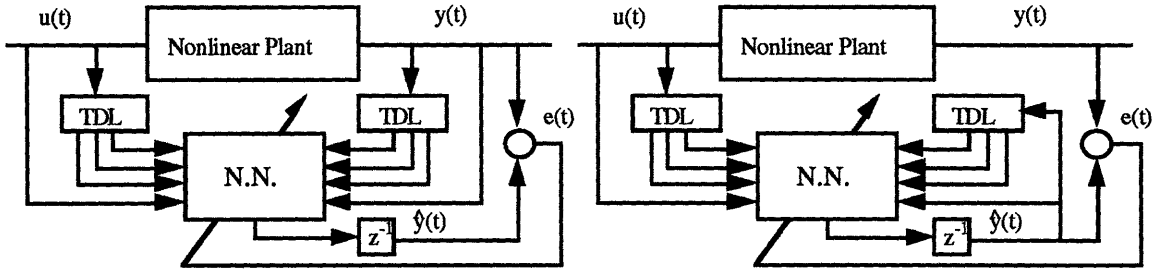
Narendra also suggested two types of identification methods. The one is a parallel identification method, which is written in the following manner for Model 4 shown above.

$$\hat{y}(t+1) = NN[\hat{y}(t), \hat{y}(t-1), \cdots, \hat{y}(t-n+1), u(t), u(t-1), \cdots, u(t-m+1)] \tag{5.3.6}$$

where $\hat{y}(t)$ is a model output, or an estimation by the model. The other method is a series-parallel identification method. This method can be written as:

$$\hat{y}(t+1) = NN[y(t), y(t-1), \cdots, y(t-n+1), u(t), u(t-1), \cdots, u(t-m+1)] \tag{5.3.7}$$

Figure 5.3.3 shows the series-parallel identification and the parallel identification.



(a) Series-parallel identification (b) Parallel identification

Fig. 5.3.3 Identification of nonlinear plants using neural network

The difference between the identification methods is that in the parallel method the outputs of the model are fed back into the model itself, and in the series-parallel method, the outputs of the plant are fed into the model. Although the configuration seems to be identical, the methods for adjusting the parameters are completely different. We can use the conventional back-propagation for the series-parallel identification. However, we cannot apply it to the parallel identification, because the adjustment of parameters by an error between the output and the desired value also causes the change of the previous outputs of the model, which are fed-back to the model.

For the problem in the parallel identification method, Narendra presented dynamic back-propagation method. This is a generalized back-propagation, which is applicable to recurrent networks that have some special configurations. Werbos presented a general method for adjusting parameters in a recurrent network. (See the book edited by White and Sofge (1992).)

Although some methods for identifying parameters in a manner of parallel identification have been proposed, they are fairy complex to implement. Even in the paper presented by Narendra and Parthasarathy (1990), where they discussed the dynamic back-propagation method, they used series-parallel identification for all the simulations. They also commented that if we use a parallel identification, there is no guarantee that the parameters will converge or that the output error will tend to zero.

5.3.3 Radial basis networks

When we use a multilayer neural network, we always face the problem of very slow convergence of the back-propagation. For example, the paper given by Narendra shows it needs more than 10000 iterations to get a satisfactory convergence for any representations

discussed above by the series-parallel identification. This is an essential drawback of these neural networks. In order to achieve faster convergence, other types of neural networks have been proposed. One of them is Radial Basis networks. The structure is also motivated by biological system. These networks consist of many spatially localized elements, and is simply described as:

$$y = \sum_{i=1}^{N} w_i R_i(u) + w_0 \tag{5.3.8}$$

where $R_i(u)$ is localized function, which usually takes a form of Gaussian function:

$$R_i(u) = \exp[-\sum_{j=1}^{n} \frac{(u_i - c_i)^2}{2\sigma_{ij}^2}] \tag{5.3.9}$$

The weights can be adjusted by a least square approach if c_i and σ_{ij} are fixed. Hence, we don't need a slow learning such as a gradient descent method. However, it needs an exponentially increased number of elements as the nonlinear system is more complicated. Therefore, the number of elements is a critical factor for describing a nonlinear system.

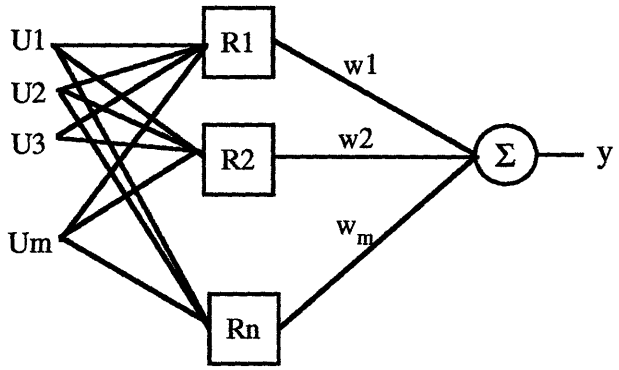


Fig. 5.3.4 Radial basis networks

5.4. Identification for linear systems and nonlinear systems

In this section, we discuss the problem of system identification. A detailed discussion for linear systems can be found in the books on this problem. (Landau (1990), Ljung (1987) and Goodwin (1984))

For nonlinear systems we deal with a minimization problem, since the identification method involves minimization problem of a quadratic cost function of the error between the outputs of the model and the given data.

The objective of this section is to know the difference between the identification for linear systems and that for nonlinear systems.

5.4.1 Linear system identification

Let us discuss how we can estimate the parameters ϑ_0 when a linear structure is given by ARMA model:

$$\begin{aligned} y(t+1) &= \alpha_0 y(t) \cdots \alpha_n y(t-n) + \beta_0 u(t) \cdots \beta_m u(t-m) \\ &= \phi(t)^t \vartheta_0 \end{aligned} \tag{5.4.1}$$

where

$$\phi(t) = [y(t), \cdots, y(t-n), u(t), \cdots, u(t-m)]$$

$$\vartheta_0 = [\alpha_0, \cdots, \alpha_n, \beta_0, \cdots, \beta_m]$$

The algorithms for on-line case and off-line case may differ due to the availability of the data set and the limit on the computation time. We are concerned with on-line algorithms for the estimation of parameters because of their direct applicability to adaptive control.

On-line estimations require the parameter estimates to be recursively updated by a small amount of measured data. What is widely used in practice has a form:

$$\hat{\vartheta}(t) = \hat{\vartheta}(t-1) + M(t-1)\phi(t-1)e(t) \tag{5.4.2}$$

where $\hat{\vartheta}(t)$: Parameter estimate

$M(t-1)$: Algorithm gain

$\phi(t-1)$: [y(t - 1),$\cdots$y(t - n), u(t - 1),,$\cdots$, u(t - m)]

$e(t)$: A modeling error

In what follows, the most fundamental algorithms are shown.

Projection algorithm

The simplest method is projection algorithm, which has a form:

$$\begin{aligned}
&\hat{\vartheta}(t) = \hat{\vartheta}(t-1) + M(t-1)\phi(t-1)e(t) \\
&M(t-1) = \frac{1}{\phi(t-1)^t \phi(t-1)} \\
&e(t) = y(t) - \phi(t-1)^t \hat{\vartheta}(t-1)
\end{aligned} \tag{5.4.3}$$

This algorithm looks for the parameters ϑ_0 so that the following cost function J is minimized:

$$J = \frac{1}{2}\left[\hat{\vartheta}(t) - \hat{\vartheta}(t-1)\right]^2 \tag{5.4.4}$$

To avoid division by zero, the algorithm gain is commonly modified as a form:

$$\begin{aligned}
&M(t-1) = \frac{a}{c + \phi(t-d)^t \phi(t-d)} \\
&c > 0 \text{ and } 0 < a < 2
\end{aligned} \tag{5.4.5}$$

The projection algorithm has the following properties.

$$\begin{aligned}
&(1)\ \left|\hat{\vartheta}(t) - \vartheta_0\right| \le \left|\hat{\vartheta}(t-1) - \vartheta_0\right| \le \left|\hat{\vartheta}(0) - \vartheta_0\right| \\
&(2)\ \lim_{N\to\infty} \sum_{t=1}^{N} \frac{e(t)^2}{c + \phi(t-1)^t \phi(t-1)} < \infty \\
&\qquad e(t) = y(t) - \phi(t-1)^t \hat{\vartheta}(t-1)
\end{aligned} \tag{5.4.6}$$

That implies the boundness of the calculation and the convergence of the parameter estimates. However, it does not necessarily guarantee their convergence to the actual values ϑ_0.

Recursive Least squares algorithm

This algorithm has a form:

$$\hat{\vartheta}(t) = \hat{\vartheta}(t-1) + M(t-1)\phi(t-d)e(t) \quad (5.4.7.a)$$

$$M(t-1) = \frac{P(t-2)}{1+\phi(t-1)^t P(t-2)\phi(t-1)} \quad (5.4.7.b)$$

$$\mathrm{P(t-1)} = \mathrm{P(t-2)} - \frac{\mathrm{P(t-2)}\phi(t-1)\phi(t-1)^t \mathrm{P(t-2)}}{1+\phi(t-1)^t P(t-2)\phi(t-1)} \quad (5.4.7.c)$$

$$p(-1) = P_0$$

where P_0 is any positive definifte matrix.

It results from the minimization of the following quadratic cost function:

$$J_N(\vartheta) = \frac{1}{2}\sum_{t=1}^{N}(y(t) - \phi(t-1)^t \vartheta(t))^2 + \frac{1}{2}(\vartheta - \hat{\vartheta}(0))^t P_0^{-1}(\vartheta - \hat{\vartheta}(0)) \quad (5.4.8)$$

This algorithm also guarantees the boundness of the computation and the convergence of the parameter estimates, but the estimated parameters do not always converge to the true values.

Persistent excitation

As mentioned above, the parameters do not necessarily converge to the true values. The following is a brief discussion of the conditions guaranteeing the convergence to the actual values.

The condition for the least squares algorithm is given by

1. The system is stable.
2. The input has a spectral distribution function which is non-zero at 2n positions or more.
3. $A(q^{-1})$ and $B(q^{-1})$ are relatively prime. (There are no common factors between both polynomials $A(q^{-1})$ and $B(q^{-1})$)

where the system is given by the DARMA model: $A(q^{-1})y(t) = B(q^{-1})u(t)$, and n is the order of $A(q^{-1})$.

Intuitively we can interpret the results above as follows.

1. Any output signals can be used because they are bounded due to the stable system, assuming the bounded inputs.
2. In order to identify a n-th order system by sinusoidal input, we need a sine wave having n different frequencies.
3. If the pole/zero cancellation takes place, the mode corresponding to it is unobservable or uncontrollable. Therefore, we cannot identify the parameters which come from unobservable/uncontrollable mode. Thus the system should be controllable and observable so that we can determine the values of all the parameters.

The input signals that make it possible to identify the true parameters are called "persistently exciting" input.

We can relax the condition **1**. The discussion about the convergence to the true parameters of an unstable system was held by Goodwin (1984). However, it is obvious that the cooling rate system is stable. Therefore we will halt our discussion here.

5.4.2 Nonlinear system identification

In this section we shall discuss how we can identify the unknown parameters in a nonlinear model. Suppose the nonlinear system is given by the general form (Multi-Input-Multi-output-mapping):

$$y = f(x,w) \qquad (5.4.9)$$

where w are the unknown parameters.

The objective of the identification is to determine the values of the unknown parameters. It is noted that we assume the structure of the nonlinear model $f(,)$ is known, but the values of the parameters w are unknown. Obviously neural networks can have this configuration.

We also note that if the nonlinear system has a special form which has a series of known nonlinear functions $\phi_n(x)$ with unknown coefficients α_n such that

$$y = \sum_{n=1}^{N} \alpha_n \phi_n(x) \qquad (5.4.10)$$

then, we can use identification algorithms for linear systems as discussed in Section 5.4.1.

The parameters are estimated so that it minimizes a cost function $J(w)$. Therefore, the identification of nonlinear systems results in the nonlinear minimization problem. As the cost function, a quadratic cost function is widely used. Some algorithms are available for minimizing nonlinear functions. However, it is quite difficult to get the solution efficiently, compared with linear case, and sometimes involves a lot of iterations to search the optimal point.

In what follows algorithms for searching the optimal points are reviewed. The detail discussion about the minimization problem is found in the book by Press et al. (1992).

(1) Cost function

The most commonly used cost function is the quadratic error:

$$E(\boldsymbol{w}) = \sum_{\mu=1}^{N} \sum_{k=1}^{K} (t_k^\mu - o_k^\mu)^2 \qquad o_k^\mu = f_k(\boldsymbol{i}^\mu, \boldsymbol{w})$$

$$\text{where } \boldsymbol{f}(\boldsymbol{i}^\mu, \boldsymbol{w}) = \begin{bmatrix} f_1(\boldsymbol{i}^\mu, \boldsymbol{w}) \\ : \\ f_K(\boldsymbol{i}^\mu, \boldsymbol{w}) \end{bmatrix}, \quad \boldsymbol{t}^\mu = \begin{bmatrix} t_1^\mu \\ : \\ t_K^\mu \end{bmatrix} \tag{5.4.11}$$

$\{\boldsymbol{i}^\mu, \boldsymbol{t}^\mu\}_{\mu=1\cdots N}$ are a set of sample data.

The task of nonlinear identification is to look for $\boldsymbol{w}$ such that it minimizes the cost function $E(\boldsymbol{w})$.

(2) Batch identification method and incremental identification method

All algorithms involved with the minimization look for the optimal parameters recursively, therefore all the sample data must be presented many times. There are two types of presentations, the batch identification method and the incremental identification method. In the batch identification method, the parameters are updated after all sample data have been presented. On the other hand, in the incremental identification method, one sample is presented, and then the parameters are updated before the next pattern is considered. Error back propagation method belongs to the latter method.

Incremental identification can be used for a nonlinear adaptive control scheme because it is suitable for on-line nonlinear identification. It can update the unknown parameters every time it measures the outputs of the plant. For off-line identification, we can use both methods.

(3) General minimization methods

We shall show some minimization methods that are commonly used. All the methods require the derivatives of the cost function.

Gradient descent method

In this method the parameters are updated by the simple rule:

$$\Delta w_i = -\eta \frac{\partial E(\boldsymbol{w})}{\partial w_i} \text{ where } \boldsymbol{w} = (w_1 \cdots w_i \cdots w_L) \tag{5.4.12}$$

where L denotes the number of the unknown parameters.
It is known that the parameters converge to a bounded region which includes a local minimum if the cost function is Lipschitz continuous. The bound of the region depends on the step size η. In order that we get a sufficiently accurate optimal point, we have to take a small step size, or diminish the step size at each iteration.

The other methods are based on the expansion of the cost function:

$$E(\boldsymbol{w}) = E(\boldsymbol{w}_0) + (\boldsymbol{w} - \boldsymbol{w}_0)\nabla E(\boldsymbol{w}_0) + \frac{1}{2}(\boldsymbol{w} - \boldsymbol{w}_0)\boldsymbol{H}(\boldsymbol{w} - \boldsymbol{w}_0) + \cdots \tag{5.4.13}$$

where $\boldsymbol{H}$ is the second derivative Hessian Matrix. Differentiating (5.4.13) and ignoring the higher-order terms in the equation gives the gradient:

$$\nabla E(\boldsymbol{w}) = \nabla E(\boldsymbol{w}_0) + \boldsymbol{H}(\boldsymbol{w} - \boldsymbol{w}_0) \tag{5.4.14}$$

Newton's method

We want to find the minimum of $E(\boldsymbol{w})$, where the condition $\nabla E(\boldsymbol{w}) = \boldsymbol{0}$ must be satisfied. Therefore, (5.4.14) yields a minimization method:

$$\boldsymbol{w}_1 = \boldsymbol{w}_0 - \boldsymbol{H}^{-1}\nabla E(\boldsymbol{w}_0) \tag{5.4.15}$$

Since we ignore the higher-order terms, $\boldsymbol{w}_1$ is used as an estimate for the location of the minimum. Therefore we look for the optimal location by using (5.4.15) iteratively, repeatedly taking $\boldsymbol{w}_1$ as the new $\boldsymbol{w}_0$. This is Newton's method.

Newton's method requires a large number of computations because we need to calculate the Hessian matrix at each iteration. If the number of the parameters is n, then we must invert a $n \times n$ matrix. It also requires the computation of the second derivatives. Therefore, Newton's method is not realistic as the actual minimization algorithm.

Most practical minimization methods use only first derivative information, combined with a line search along a selected direction. Starting from $\boldsymbol{w}_0$ with direction $\boldsymbol{d}$ the search is made on the line:

$$\boldsymbol{w} = \boldsymbol{w}_0 + \lambda \boldsymbol{d} \tag{5.4.16}$$

and choose λ to minimize the cost function. When we reach the point which minimizes the cost function, we again set up another direction. The line search is repeated until the cost function no longer decreases.
The following methods have been proposed for giving the direction $\boldsymbol{d}$.

Steepest descent method

The simplest approach using line search is steepest descent method, where the direction is given by:

$$\boldsymbol{d} = -\nabla E(\boldsymbol{w}) \tag{5.4.17}$$

This method sometimes brings about redundant small steps. (Press et al. (1992)) Therefore the method is not considered as a good algorithm for the line search.

Conjugate gradient method

The idea is based on the approximation of the nonlinear function by a quadratic form. The vectors $\boldsymbol{d}_i$ and $\boldsymbol{d}_j$ which satisfy

$$\boldsymbol{d}_i H \boldsymbol{d}_j = 0 \tag{5.4.18}$$

are said to be conjugate, and we can find n independent and conjugate vectors when the dimension of $\boldsymbol{H}$ is n. Suppose $\boldsymbol{H}$ is a constant matrix. We would like to minimize a quadratic form

$$E(\boldsymbol{w}) = \boldsymbol{c} + (\boldsymbol{w} - \boldsymbol{w}_0)\boldsymbol{b} + \frac{1}{2}(\boldsymbol{w} - \boldsymbol{w}_0)\boldsymbol{H}(\boldsymbol{w} - \boldsymbol{w}_0) \tag{5.4.19}$$

where $\boldsymbol{b}$ and $\boldsymbol{c}$ are constant vectors.

If we use conjugate vectors as the directions of the line search, we can reach the optimal point which minimizes the quadratic form by n line searches.
The conjugate vectors can be constructed by the successive manipulation

$$\begin{aligned} \boldsymbol{d}_{i+1} &= \boldsymbol{g}_{i+1} + \frac{\boldsymbol{g}_{i+1} \cdot \boldsymbol{g}_{i+1}}{\boldsymbol{g}_i \cdot \boldsymbol{g}_i}\boldsymbol{d}_i \qquad \text{(Fletcher-Reeves method)} \qquad (5.4.20) \\ \boldsymbol{g}_{i+1} &= -\nabla E(\boldsymbol{w}_i) \end{aligned}$$

where $\boldsymbol{w}_i$ is the optimal point at the i-th line minimization.

The conjugate gradient method directly applies the algorithm (5.4.20) to the general minimization problem. By this method we can avoid redundant small steps, which is seen in the steepest descent method. Instead of (5.4.20), the algorithm

$$\boldsymbol{d}_{i+1} = \boldsymbol{g}_{i+1} + \frac{(\boldsymbol{g}_{i+1} - \boldsymbol{g}_i) \cdot \boldsymbol{g}_{i+1}}{\boldsymbol{g}_i \cdot \boldsymbol{g}_i}\boldsymbol{d}_i \qquad \text{(Polak-Ribiere method)} \qquad (5.4.21)$$

is sometimes preferred. This algorithm is essentially equal to the steepest descent method when $\boldsymbol{g}_{i+1} = \boldsymbol{g}_i$.

If the cost function takes the form in (5.4.19), it reaches the minimum in exactly n steps. However, for the minimization of the general cost function, we need more iterations because the above method simply approximates the cost function by the form in (5.4.19).

Quasi-Newton's method

Here the idea is to use Newton's rule (5.4.15) with an approximation to the inverse Hessian $\boldsymbol{H}^{-1}$. This is achieved through a rather complicated iterative scheme

$$\begin{aligned} \boldsymbol{w}_{i+1} &= \boldsymbol{w}_i - \lambda \boldsymbol{G}_{i+1} \nabla E(\boldsymbol{w}_i) \\ \boldsymbol{G}_{i+1} &= \boldsymbol{G}_i + \frac{(\boldsymbol{w}_{i+1} - \boldsymbol{w}_i) \otimes (\boldsymbol{w}_{i+1} - \boldsymbol{w}_i)}{(\boldsymbol{w}_{i+1} - \boldsymbol{w}_i) \cdot (\nabla E(\boldsymbol{w}_{i+1}) - \nabla E(\boldsymbol{w}_i))} \\ &\quad - \frac{[\boldsymbol{G}_i(\nabla E(\boldsymbol{w}_{i+1}) - \nabla E(\boldsymbol{w}_i))] \otimes [\boldsymbol{G}_i(\nabla E(\boldsymbol{w}_{i+1}) - \nabla E(\boldsymbol{w}_i))]}{(\nabla E(\boldsymbol{w}_{i+1}) - \nabla E(\boldsymbol{w}_i)) \cdot \boldsymbol{G}_i \cdot (\nabla E(\boldsymbol{w}_{i+1}) - \nabla E(\boldsymbol{w}_i))} \end{aligned} \tag{5.4.21}$$

where $\otimes$ denotes the outer product of two vectors, λ is chosen by line minimization, and $\boldsymbol{G}$ corresponds to the approximation of the Hessian matrix.

Among the algorithms above, the gradient descent method shows the slowest convergence, while the conjugate gradient method and the quasi-Newton's method are considered to be the fastest algorithm.

(4) Algorithm for the incremental method

As mentioned above, an incremental method updates the parameters every time a sample of data is presented. The set of sample data may comprise an infinite number of data. Therefore, we may not precisely define the whole set of sample data. Line minimization methods are not available for this problem.

The gradient descent method is widely used as the algorithm for the incremental method. In this case the parameters are updated by:

$$\Delta \boldsymbol{w}_i = -\eta \frac{\partial E_\mu(\boldsymbol{w})}{\partial \boldsymbol{w}} \tag{5.4.22}$$

where $E_\mu(\boldsymbol{w})$ is the cost function at the μ-th sample, which is usually given by:

$$E_\mu(w) = \sum_{k=1}^{K} (t_k^\mu - o_k^\mu)^2 \qquad o_k^\mu = f_k(i^\mu, w) \tag{5.4.23}$$

$\{i^\mu, t^\mu\}$ are a sample data.

This method is used as error back-propagation method in the neural network computation.

Obviously we need make the step-size smaller than the general minimization case shown in (5.4.12) so that we can get satisfactory convergence.

5.4.3 Some comments on nonlinear identification

(1) Comparison between recursive least square algorithm and incremental gradient descent method

We have so far discussed the identification methods for linear systems, and for nonlinear systems. Of course we can apply the nonlinear identification methods to the linear system case. However, the identification methods for nonlinear systems are slower than those for linear systems.

Consider the recursive least square algorithm and the incremental gradient descent method. The objectives of both methods are to minimize the cost function, the square error of the output. Both methods update the unknown parameters every time a sample is given. It seems both methods are parallel to each other. However, the performance of the two methods is quite different. Figure 5.4.1 shows how fast both methods estimate the parameters. In the figure, $y(t+1) = 0.5y(t) + 0.5u(t)$ is used as the plant model, and a pure sinusoidal wave is given as the persistently exciting input. We also take 0.3 as the step size of the gradient descent method, which is typically used. We can see the incremental method is 5 times slower than the recursive least square algorithm.

If we use the neural network as the model describing the linear system, the problem would be more complicated, and the convergence of the parameter estimation would be further deteriorated.

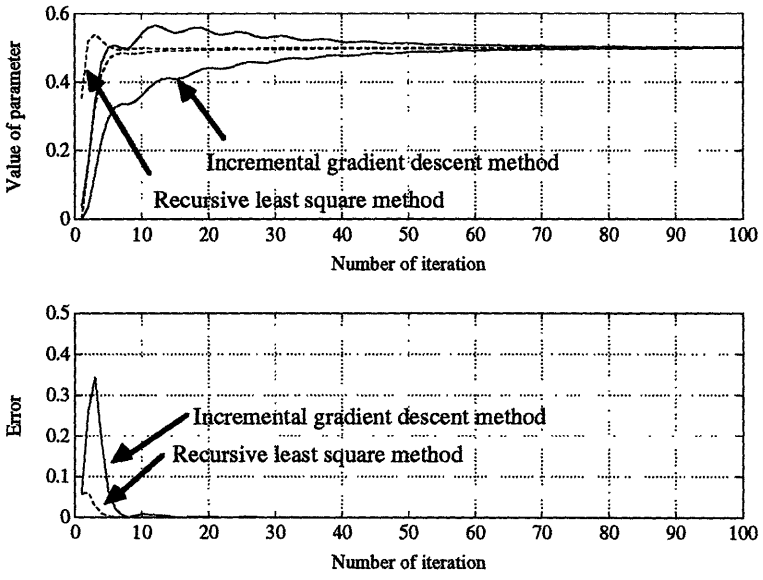


Fig. 5.4.1 Comparison between incremental gradient method and recursive least square method

(2) Incremental identification method and batch identification method

When we consider the performance of nonlinear minimization, we have to pay attention to two points:

(1) The speed of convergence

(2) The problem of local minimum

As mentioned above, if we use the incremental gradient descent method, we have to take a quite small step size to get satisfactory convergence. Therefore, the batch method based on line minimization generally show quite faster convergence than the incremental method.

However, we also have to be concerned with the location where the parameters converge. All the methods discussed above do not necessarily guarantee the convergence to the global minimum. There is a possibility that the parameters converge to a local minimum, and are trapped there. In particular, for neural network computations, the geometry of the cost function is quite complicated, involving a large number of hills and valleys.

When we use the batch identification based on line minimization, they are easily trapped by a local minimum, because these methods try to find the exact minimum location around the given initial location.

However, by the incremental method, we can sometimes get out of the local minimum if it is not so deep. Consider how parameters are updated by the incremental method. The cost of a whole set of sample data would slowly decrease, fluctuating by the update, because the incremental method updates the parameters by a simple method in (5.4.22). The update by a pair of sample data does not always reduce the cost of a whole set of sample data, although the cost would decrease after completing the updates all over the pairs of sample data. The fluctuation could give an effect of random noise for the minimization. It is well known the noise sometimes helps it to get out of the valley with a shallow depth. That is the reason the incremental method reduces the possibly of unfavorable trapping in local minima. Therefore, we may choose to use the incremental method even if it shows a slower convergence.

5.5 Control dealing with time-delay

As mentioned in Chapter 4, we must also cope with the problem of a time-delay to implement the control system for the cooling rate.

Although a time-delay is common in process control, there is little literature that deals precisely with a time-delay system. A time-delay T is sometimes treated as a unmodeled factor, or approximated as a first order or a second order transfer function:

$$\frac{1-sT/2}{1+sT/2} \qquad \text{(First order Padé approximate)}$$

$$\frac{1-sT/2+(sT)^2/12}{1+sT/2+(sT)^2/12} \qquad \text{(Second order Padé approximate)} \tag{5.5.1}$$

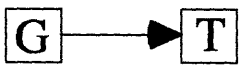
However, such approximation does not work for a large time-delay system.

As a practical method coping with a time-delay, Smith's principle is discussed in the sequel (Refer to Marshall (1978)).

5.5.1 Smith's principle

The Smith's principle is based on the assumption:

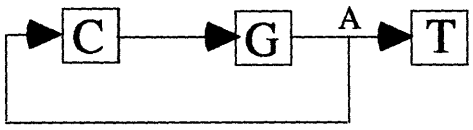
(1) The plant dynamics has a cascade structure comprising a delay-free part G and a pure time-delay part T as shown below.



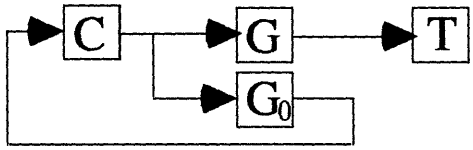
(2) An exact model describing the delay-free part G_0 and the pure delay system T_0 is available.

Note that the term G just shows the input-output dynamic relation, and is not necessarily a transfer function so that we can accommodate a nonlinear dynamics. We also note the locations of G and T are exchangeable. That means we do not care about where the delay takes place.

Suppose we design a controller C which satisfies a given performance in the manner shown below.



However, it is impossible to make a feed-back loop in the figure due to the inaccessibility of point A. Therefore, we will use the model G_0 instead of G. Note that we can completely ignore the effect of the time-delay when we design the controller.



The above method does not include any feed-back from the output of the actual plant. Therefore, we place a feed-back from the output to eliminate the difference between the output of the plant and that of the model. That is Smith's principle. Figure 5.5.1 shows the final configuration of Smith's principle.

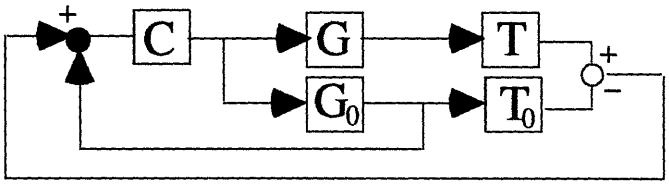


Fig. 5.5.1 Smith's principle

This idea has some advantages for designing a controller.

1. The principle can be applied any dynamics even if it is nonlinear.

2. We can design the controller, completely ignoring the effect of the time-delay.

The controller controls the plant, based on the model-based prediction after the time-delay. Therefore, it is essential to get the exact model for the delay-free part and the time-delay part.

We can raise many questions about the stability, the disturbance rejection, and the insensitivity to model perturbation and sensor noise. There are, therefore, many critics

against this principle. However, we can also find many reports about successful applications by this principle. It deeply depends on the nature of the plant and the good modeling whether this method works well or not. We can find an extensive discussion of Smith's method in the book by Marshal (1978).

5.5.2 Other methods for time-delay

Doyle et al. (1992) discussed how we can accommodate a time-delay by the conventional design method for continuos-time systems from the view-point of robust control. The controller design for discrete-time systems can also easily deal with a time-delay. Since these controllers accommodate simply by a conventional feed-back control, the performance would not be necessarily satisfactory.

As we shall see in Section 5.7.2, one-step ahead predictive control can successfully cope with a time-delay system by employing a control law based on a prediction.

5.6 Feed-back control for nonlinear systems

This section summarizes nonlinear controllers, which requires discussion of mathematical tools such as Lyapunov function, Lie algebra, etc. However, in this summary we will just show the concept of the controls, and in particular we pay attention to their applicability to the cooling rate control. We can refer to the detailed discussion in the book by Slotine and Li. (1991).

5.6.1 Feed-back linearization

The idea of feed-back linearization is to first transform a nonlinear system into a linear system, then use the well-known linear design techniques. Note that this method differs from conventional linearization (Jacobian linearization).

Feed-back linearization has been well investigated for continuous-time systems. (Isidori (1989), Vidyasagar (1993)) Feed-back linearization for discrete-time systems are still under research. (See the papers by Grizzle (1986) and Jakubczyk (1987)), therefore, first we discuss a continuous-time case.

In this method it is assumed that a nonlinear system is given by the state space equation:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})u(t) \\ y &= h(\boldsymbol{x}) \end{aligned} \tag{5.6.1}$$

Consider the following differentiation of y.

$$\begin{aligned} \dot{y} &= L_f \boldsymbol{h}(\boldsymbol{x}) + L_g \boldsymbol{h}(\boldsymbol{x})u \\ \ddot{y} &= L_f^2 \boldsymbol{h}(\boldsymbol{x}) + L_g L_f \boldsymbol{h}(\boldsymbol{x})u \\ \dddot{y} &= L_f^3 \boldsymbol{h}(\boldsymbol{x}) + L_g L_f^2 \boldsymbol{h}(\boldsymbol{x})u \\ &\vdots \\ y^{(r)} &= L_f^r \boldsymbol{h}(\boldsymbol{x}) + L_g L_f^{r-1} \boldsymbol{h}(\boldsymbol{x})u \end{aligned} \tag{5.6.2}$$

where L denotes the Lie derivative.
and suppose

$$L_g L_f^{i-1} h(\boldsymbol{x}) \begin{cases} = 0 & (i < r) \\ \neq 0 & (i = r) \end{cases} \quad \boldsymbol{x} \in \Omega_x \tag{5.6.3}$$

where Ω_x is the region of interest.

Then if the control law:

$$u = \frac{1}{L_g L_f^{r-1} h(\boldsymbol{x})} (-L_f^r h(\boldsymbol{x}) + y_d^{(r)} - \alpha_1 e \cdots - \alpha_{r-1} e^{(r-1)})$$
$$e = y_d - y \tag{5.6.4}$$

where y_d is the reference input.

is applied to the system, the closed-loop system is given by:

$$y^{(r)} = y_d^{(r)} - \alpha_1 e \cdots - \alpha_{r-1} e^{(r-1)} \tag{5.6.5}$$

Therefore, we can select the parameters $\alpha_1, \cdots, \alpha_{r-1}$ so that the closed system is stable and shows a desirable response. Thus, we can achieve the stable nonlinear control. Note the system given by (5.6.5) is a linear system. Therefore we can use powerful design methods for linear systems.

The validity of the above discussion is more complicated and requires more mathematics. Here we would like to pay attention to the requirements to achieve the feed-back linearization. From the equation (5.6.4), it requires

1. The full state has to be measured.
2. The internal dynamics, the dynamics that is hidden within the system by the feed-back linearization, must be stable. (Suppose r is less than the order of the open loop system. Then there must be an internal dynamics hidden by the feed-back linearization.)
3. The model should be explicitly presented as a differentiable nonlinear function.

In our problem we only know the dynamic relation between the input and the output. Furthermore the relationship is given by a parameterized nonlinear mapping when we use neural networks. Therefore, the applicability to our problem diminishes to the following simpler case.

If the system is given by a simpler form:

$$
\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= x_3 \\
&\vdots \\
\dot{x}_{n-1} &= x_n \\
\dot{x}_n &= f(\boldsymbol{x}) + g(\boldsymbol{x})u \\
y &= x_1
\end{aligned}
\tag{5.6.6}
$$

the control is simply given by

$$
u = \frac{1}{g(\boldsymbol{x})}(y_d^{(n)} - \alpha_1 e \cdots - \alpha_{n-1} e^{(n-1)} - f(\boldsymbol{x})) \tag{5.6.7}
$$

Since the system in (5.6.6) is equivalent to

$$
y^{(n)} = f(y, \dot{y}, \cdots, y^{(n-1)}) + g(y, \dot{y}, \cdots, y^{(n-1)})u(t) \tag{5.6.8}
$$

we have only to design the control law:

$$
u = \frac{1}{g(y, \cdots, y^{(n-1)})}(y_d^{(n)} - \alpha_1 e \cdots - \alpha_{n-1} e^{(n-1)} - f(y, \cdots, y^{(n-1)})) \tag{5.6.9}
$$

More generally if the system is given by:

$$
y^{(n)} = f(y, \cdots, y^{(n-1)}, u) \tag{5.6.10}
$$

then the feed-back linearization leads to the problem we look for the function $h(y, \cdots, y^{(n-1)}, v)$ such that

$$
v = f(y, \cdots, y^{(n-1)}, h(y, \cdots, y^{(n-1)}, v)) \qquad \forall v, y^{(i)} \quad (i = 0 \cdots n-1) \tag{5.6.11}
$$

Using the function $h(y, \cdots, y^{(n-1)}, v)$, our task results in designing the control law such that

$$
u = h(y, \cdots, y^{(n-1)}, y_d^{(n)} - \alpha_1 e \cdots - \alpha_{n-1} e^{(n-1)}) \tag{5.6.12}
$$

Obviously the control law above yields

$$y^{(n)} = y_d^{(n)} - \alpha_1 e \cdots - \alpha_{r-1} e^{(n-1)} \tag{5.6.13}$$

Now let us go back to the feed-back linearization of discrete-time systems. Considering the discussion above, we assume the system is given by:

$$y(t+1) = f(y(t), \cdots, y(t-n), u(t)) \tag{5.6.14}$$

Then, first we look for the function $h(y(t), \cdots, y(t-n), v)$ such that

$$v = f(y(t), \cdots, y(t-n), h(y(t), \cdots, y(t-n), v)) \tag{5.6.15}$$

Next we will design the control law:

$$u = h(y(t), \cdots, y(t-n), y_d(t+1) - \alpha_0 e(t) \cdots - \alpha_n e(t-n)) \tag{5.6.16}$$

We have so far discussed the concept of the feed-back linearization. From the discussion we can see the following limitations of this method.

1. It requires a full state feed-back. If it is not available, the plant must be limited to the system described by (5.6.14).
2. No robustness is guaranteed. As it was seen, the concept of the feed-back linearization lies in the complete cancellation of the nonlinear dynamics. Therefore, the model error would lose the stability and degrade the performance.

Feed-back linearization can not treat a plant described as:

$$y(t+1) = f(y(t), \cdots, y(t-n), u(t), \cdots, u(t-m)) \tag{5.6.17}$$

One may claim we can apply a similar idea by looking for the function $h(y(t), \cdots, y(t-n), u(t-1), \cdots, u(t-m), v)$ which satisfies

$$v = f(y(t), \cdots, y(t-n), h(y(t), \cdots, y(t-n), u(t-1), \cdots, u(t-m), v), u(t-1), \cdots, u(t-m)) \tag{5.6.18}$$

Such concept leads to one-step-ahead predictive control, which will be discussed in the next section.

5.6.2 One-step-ahead predictive control

One-step-ahead control scheme for linear systems has been well investigated by Goodwin and Sin (1984). The summary for linear systems is based on his book. The idea of this scheme can be extended to nonlinear systems.

First we discuss the linear case, and then we shall consider how we can apply the concept to nonlinear systems. This predictive control is easily extended to adaptive control together with estimation algorithms.

(1) One-step-ahead predictive control for SISO linear systems

In this section it is assumed that the system is given by a difference operator representation, or ARMA model:

$$
\begin{aligned}
&A(q^{-1})y(t) = B(q^{-1})u(t) \\
&\text{where} \\
&A(q^{-1}) = 1 + a_1 q^{-1} + \cdots + a_n q^{-n} \\
&B(q^{-1}) = q^{-d}(b_0 + b_1 q^{-1} + \cdots + b_{n_1} q^{-n_1}) \qquad b_0 \neq 0
\end{aligned}
\tag{5.6.19}
$$

This ARMA model can be expressed in d step-ahead predictor form as

$$
\begin{aligned}
&y(t+d) = \alpha(q^{-1})y(t) + \beta(q^{-1})u(t) \\
&\text{where} \\
&\alpha(q^{-1}) = \alpha_0 + \alpha_1 q^{-1} + \cdots + \alpha_{n-1} q^{-(n-1)} \\
&\beta(q^{-1}) = \beta_0 + \beta_1 q^{-1} + \cdots + \beta_{n_1+d-1} q^{-(n_1+d-1)} \qquad \beta_0 \neq 0
\end{aligned}
\tag{5.6.20}
$$

The feed-back control law for the system is given by a simple manipulation as shown below.

Prototype one-step-ahead control law

The feed-back control law which brings the output at time $t+d$, $y(t+d)$, to the desired bounded value $y_d(t+d)$ is given by

$$u(t) = \frac{y_d(t+d) - \alpha(q^{-1})y(t) - \beta'(q^{-1})u(t)}{\beta_0}$$

where (5.6.21)

$$\alpha(q^{-1}) = \alpha_0 + \alpha_1 q^{-1} + \cdots + \alpha_{n-1} q^{-(n-1)}$$

$$\beta'(q^{-1}) = \beta_1 q^{-1} + \cdots + \beta_{n_1+d-1} q^{-(n_1+d-1)} \qquad \beta_0 \neq 0$$

That is the prototype one-step-ahead control. The resulting closed-loop system has dynamics described by

$$\begin{aligned} y(t) &= y_d(t) && t \geq d \\ B(q^{-1})u(t) &= A(q^{-1})y_d(t) && t \geq d+n \end{aligned} \tag{5.6.22}$$

Equation (5.6.22) shows the output is bounded, and the input is also bounded provided that the "inverse" model $z^d B(z^{-1})/A(z^{-1})$ is stable. (If a pole /zero cancellation takes place, the unobservable modes must be stable.) We can find more precise conditions for the boundness of the input and the output (Goodwin and Sin(1984)).

Note that here we assume that we have the model exactly same as the actual plant. If the model has an error from the actual process, the above discussion about the boundness would be degraded. Next, the control law calls for an excessive effort to bring the output $y(t+d)$ to the desired value $y_d(t+d)$ in one step. This would result in an unfavorable saturation of the input and the excitement of unmodeled resonance. Goodwin also presents other configuration for the control law to avoid the latter drawbacks above. The prototype one-step-ahead control minimizes the cost function:

$$J(t+d) = \frac{1}{2}(y(t+d) - y_d(t+d))^2 \tag{5.6.23}$$

To make a constraint on the input, we consider the following cost function:

$$J(t+d) = \frac{1}{2}\{(y(t+d) - y_d(t+d))^2 + \lambda u(t)^2\} \tag{5.6.24}$$

The minimization of this cost function leads to the weighted one-step-ahead control.

Weighted one-step-ahead control

The control law is given by:

$$u(t)=\frac{\beta_0\left\{y_d(t+d)-\alpha(q^{-1})y(t)-\beta'(q^{-1})u(t)\right\}}{\beta_0^2+\lambda} \tag{5.6.25}$$

Then the closed-loop system has the following dynamics

$$\begin{aligned}&[\beta(q^{-1})+\frac{\lambda}{\beta_0}a(q^{-1})]y(t+d)=q^dB(q^{-1})y_d(t+d)\\&[\beta(q^{-1})+\frac{\lambda}{\beta_0}a(q^{-1})]u(t)=A(q^{-1})y_d(t)\end{aligned} \tag{5.6.26}$$

The closed system should be stable so that we can get the bounded output and input.

The above closed loop system can not always be stabilized because the system has only one degree of freedom. In order to enhance the degrees of freedom we can have the other control law, which minimizes the cost function:

$$\begin{aligned}&J(t+d)=\frac{1}{2}[y(t+d)-y_d(t+d)+\frac{\lambda}{2}\bar{u}(t)^2]^2\\&\text{where } P(q^{-1})\bar{u}(t)=R(q^{-1})u(t)\\&P(q^{-1})=1+p_1q^{-1}+\cdots+p_lq^{-l}\\&R(q^{-1})=1+r_1q^{-1}+\cdots+r_lq^{-l}\end{aligned} \tag{5.6.27}$$

Then, the control law is given by:

$$\begin{aligned}&u(t)=\frac{\beta_0\left\{y_d(t+d)-\alpha(q^{-1})y(t)-\beta'(q^{-1})u(t)\right\}+\lambda(P'(q^{-1})\bar{u}(t)-R'(q^{-1})u(t))}{\beta_0^2+\lambda}\\&\text{where } P'(q^{-1})=p_1q^{-1}+\cdots+p_lq^{-l} \text{ and } R'(q^{-1})=r_1q^{-1}+\cdots+r_lq^{-l}\end{aligned} \tag{5.6.28}$$

Using this configuration we can stabilize any systems.

Pole assignment controller

In the method discussed above we cannot say precisely where the closed-loop poles are located. Here we assign the poles to desired locations while retaining the predictive control structure. Conceptually the approach differs from the methods above.

First we consider the general control law:

$$L(q^{-1})u(t) = -P(q^{-1})y(t) + M(q^{-1})y_d(t+d) \tag{5.6.29}$$

for the plant:

$$A(q^{-1})y(t) = B(q^{-1})u(t) \tag{5.6.30}$$

(5.6.29) and (5.6.30) yield the following closed-loop system.

$$\begin{aligned} &[L(q^{-1})A(q^{-1}) + q^{-d}B'(q^{-1})P(q^{-1})]y(t) = B'(q^{-1})M(q^{-1})y_d(t) \\ &\text{where } B(q^{-1}) = q^{-d}B'(q^{-1}) \end{aligned} \tag{5.6.31}$$

The idea is to make the polynomial $L(q^{-1})A(q^{-1}) + q^{-d}B'(q^{-1})P(q^{-1})$ in (5.6.31) be equal to a desired closed-loop polynomial $A_d(q^{-1})$. The method is supported by the fact we can determine unique polynomials $L(q^{-1})$ and $M(q^{-1})$, if $A(q^{-1})$ and $B(q^{-1})$ are relatively prime and $A_d(q^{-1})$ is any arbitrary polynomial of degree $(2n-1)$, where n is the max degree of $A(q^{-1})$ and $B(q^{-1})$. (See the book by Goodwin.)

The design resorts to the state space representation of the system, and we use the observer-based state feed-back for the pole assignment. (See, for example, Franklin and Powell (1990)) Therefore, the control algorithm is rather complicated.

The procedure for the design of one-step-ahead control is as follows.

1. Convert the ARMA model to the predictor form:

$$
\begin{aligned}
&y(t+d) = \alpha(q^{-1})y(t) + \beta(q^{-1})u(t) \\
&\text{where} \\
&\alpha(q^{-1}) = \alpha_0 + \alpha_1 q^{-1} + \cdots + \alpha_{n-1} q^{-(n-1)} \\
&\beta(q^{-1}) = \beta_0 + \beta_1 q^{-1} + \cdots + \beta_{n_1+d-1} q^{-(n_1+d-1)} \qquad \beta_0 \neq 0
\end{aligned}
\tag{5.6.32}
$$

2. Specify the cost function.

3. Design the control law based on the cost function, using the predictor form.

(2) One-step-ahead predictive control for SISO nonlinear systems

The concept of one-step-ahead predictive control can be applied to nonlinear systems. Suppose we have a nonlinear predictor form which is given by:

$$
y(t+d) = f(y(t),\cdots,y(t-n),u(t),\cdots,u(t-m)) \tag{5.6.33}
$$

If we can get the function $h(y(t),\cdots,y(t-n),u(t-1),y_d(t+d))$ such that

$$
\begin{aligned}
&y_d(t+d) = \\
&f(y(t),\cdots,y(t-n),h(y(t),\cdots,y(t-n),u(t-1),\cdots,u(t-m),y_d(t+d)),\cdots,u(t-m))
\end{aligned}
\tag{5.6.34}
$$

the control law in the prototype one-step-ahead predictive control for nonlinear systems can be simply given by:

$$
u(t) = h(y(t),\cdots,y(t-n),u(t-1),\cdots,u(t-m),y_d(t+d)) \tag{5.6.35}
$$

although we can not say anything about the stability of the system. The control law must minimize the cost function

$$
J = \frac{1}{2}(y(t+d) - y_d(t+d))^2 \tag{5.6.36}
$$

This method raises some questions about the stability of the closed system, the existence of the function $h()$, and how we can get the function $h()$.

Intuitively, if the obtained function $h()$ always generates a bounded control signal, the closed loop system is bounded, supposing the plant is stable. If not, this method does not work. Obviously if the plant is unstable, the method cannot stabilize the system.

If we use neural networks for representing the function $f()$ in equation (5.6.31), we can also approximate the function $h()$ by neural networks. The method to get the function $h()$ is simply to minimize the cost function in the manner:

$$J = \sum_i \left[f(y_1^i(t), \cdots, y_n^i, h(y_1^i, \cdots, y_n^i, u_2^i, \cdots, u_m^i, v^i), \cdots, u_m^i, v^i) - v^i \right]^2 \tag{5.6.37}$$

where $\{y_1^i, \cdots, y_n^i, u_2^i, \cdots, u_m^i, v^i\}_{i=1 \cdots N}$ are a set of sample data.

For the weighted one-step-ahead control, similarly we look for the function $h()$ which minimizes the cost function:

$$J = \sum_i \left[\left\{ f(y_1^i(t), \cdots, y_n^i, h(y_1^i, \cdots, y_n^i, u_2^i, \cdots, u_m^i, v^i), \cdots, u_m^i, v^i) - v^i \right\}^2 + \lambda h(y_1^i, \cdots, y_n^i, u_2^i, \cdots, u_m^i, v^i)^2 \right] \tag{5.6.38}$$

Then, the control law is given by:

$$u(t) = h(y(t), \cdots, y(t-n), u(t-1), y_d(t+d)) \tag{5.6.39}$$

If the minimization of the cost function is executed on-line, it leads to a nonlinear adaptive scheme. Adaptive control using neural networks will be discussed in Section 5.8.

(3) Comments on one-step-ahead controller

Predictor form

If the system involves a rather long time-delay, the predictor form would be complicated, involving a large number of unknown parameters. For example, consider a first order system with the time-delay of ten times of the sample interval. Then the ARMA model is given by

$$(1-\alpha q^{-1})y(t) = q^{-11}(1-\alpha)u(t)$$

The above equation corresponds to

$$y(t+10) = \alpha y(t+9) + (1-\alpha)u(t-1)$$

Since $y(t+9) = \alpha y(t+8) + (1-\alpha)u(t-2)$,

$$\begin{aligned} y(t+10) &= \alpha\{\alpha y(t+8) + (1-\alpha)u(t-2)\} + (1-\alpha)u(t-1) \\ &= \alpha^2 y(t+8) + (1-\alpha)u(t-1) + \alpha(1-\alpha)u(t-2) \end{aligned}$$

Similar manipulations yield

$$\begin{aligned} y(t+10) &= \alpha\{\alpha y(t+8) + (1-\alpha)u(t-2)\} + (1-\alpha)u(t-1) \\ &= \alpha^{10} y(t) + (1-\alpha)\sum_{j=1}^{10} \alpha^{j-1} u(t-j) \end{aligned}$$

Therefore even if an ARMA model is simple, the predictor form becomes complicated. Similarly, if it has higher order terms, then the corresponding predictor form becomes more complicated.

Comparison with Smith's principle

Control by Smith's principle is also based on the prediction. We would like to discuss the difference between both controllers by Smith's principle and by one-step-ahead control.

Smith's method uses only the values of the input given to the plant to predict the output after the time-delay. That is based on the fact the prediction error converges to zero, if the system is linear and stable, and the model is exactly same as the actual plant. Consider the system and the predictor given by

$$\begin{aligned} A(q^{-1})y(t) &= B(q^{-1})u(t) \qquad \text{(Actaul plant)} \\ A(q^{-1})\hat{y}(t) &= B(q^{-1})u(t) \qquad \text{(Predictor)} \end{aligned} \tag{5.6.37}$$

Then

$$A(q^{-1})e(t) = 0 \qquad \text{where } e(t) = \hat{y}(t) - y(t) \tag{5.6.38}$$

Therefore, If the system is stable, the error $e(t)$ converges to zero. That implies that the error at the initial stage disappears with time. For nonlinear systems, we could say a similar result, but we would need further analysis and discussion.

The one-step-ahead control is based on the d step-ahead prediction. Therefore, the predictor has only to predict the future with some accuracy.

The difference between both methods is that Smith's method requires a model exactly same as the plant model, while in one-step-ahead control we just need a model which can predict d-step-ahead plant output with some accuracy. Thus, the model for Smith's method should be more accurate than that for one-step-ahead predictive control.

The discussion above shows a disadvantage of Smith's principle. However it has the advantage that we don't need the complicated predictor form. We just need an ARMA model. Therefore, the choice of the method depends on the complexity of the system, and how accurate a model we can obtain.

5.6.3 Modifications of nonlinear one-step-ahead predictive control

In this section let us consider two types of modifications of nonlinear one-step ahead control:

Feed-back linearization-like control

Generic Model control.

Feed-back linearization-like control

The idea here is to make a controller by the method similar to feed-back linearization. The objective of this control is to make the output asymptotically converge to the desired output.

The manipulation is almost identical to the discussion in Section 5.6.1. For linear systems, we use the control law:

$$u(t) = \frac{y_d(t+d) - \alpha(q^{-1})y(t) - \beta'(q^{-1})u(t) - \kappa(q^{-1})e(t)}{\beta_0}$$

where

$$\begin{aligned} &\alpha(q^{-1}) = \alpha_0 + \alpha_1 q^{-1} + \cdots + \alpha_{n-1} q^{-(n-1)} \\ &\beta'(q^{-1}) = \beta_1 q^{-1} + \cdots + \beta_{n_1+d-1} q^{-(n_1+d-1)} \qquad \beta_0 \neq 0 \\ &\kappa(q^{-1}) = \kappa_0 + \kappa_1 q^{-1} + \cdots + \kappa_l q^{-(l-1)} \\ &e(t) = y(t) - y_d(t) \end{aligned} \tag{5.6.39}$$

instead of (5.6.21). For nonlinear systems, we can apply the control law:

$$\begin{aligned} &u(t) = h(y(t), \cdots, y(t-n), u(t-1), \cdots, u(t-m), \bar{y}(t)) \\ &\bar{y}(t) = y_d(t+d) - \kappa_0 e(t) - \kappa_1 e(t-1) \cdots - \kappa_l e(t-l) \end{aligned} \tag{5.6.40}$$

instead of (5.6.32).

Both control laws result in

$$e(t+d) + \kappa_0 e(t) + \kappa_1 e(t-1) \cdots + \kappa_l e(t-l) = 0 \tag{5.6.41}$$

If the system (5.6.41) is stable, the error converges to zero with time. For linear systems the control is stable, if the condition of one-step-ahead control is satisfied. However, for nonlinear systems, we need further analysis.

Generic Model Control

As a practical nonlinear control method, Lee and his colleagues have proposed Generic Model Control (1988). The idea is quite similar to feed-back linearization. They first assume the structure of the system is given by

$$\dot{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{y},\boldsymbol{u},\boldsymbol{d},t,\vartheta) \tag{5.6.42}$$

for continuos-time multivariable systems. In this equation $\boldsymbol{y}$ is a vector of process outputs, $\boldsymbol{u}$ is a vector of process inputs, $\boldsymbol{d}$ is a vector of process disturbances, t is time and ϑ is a vector of model parameters and $\boldsymbol{f}()$ is a nonlinear known function.

They claim the control signal $\boldsymbol{u}$ can be determined by solving the equation:

$$\boldsymbol{f}(\boldsymbol{y},\boldsymbol{u},\boldsymbol{d},t,\vartheta) = K_1(y_d - y) + K_2\int_0^t (y_d - y)dt \tag{5.6.43}$$

where K_1 and K_2 are the gain matrices.

It seems to be PI controller for nonlinear systems.

Let us consider the case for SISO discrete-time system given by:

$$y(t+1) = f(y(t),\cdots,y(t-n),u(t),\cdots,u(t-m)) \tag{5.6.44}$$

then, the control law is determined by the solution $u(t)$ which satisfies:

$$\begin{aligned} &f(y(t),\cdots,y(t-n),u(t),\cdots,u(t-m)) = v(t) \\ &v(t) = v(t-1) + k_1(e(t) - e(t-1)) + k_2 e(t) \\ &e(t) = y_d(t) - y(t) \end{aligned} \tag{5.6.45}$$

To get the solution, we can use $h()$ in (5.6.31) again. Then the control law is given by

$$u(t) = h(y(t),\cdots,y(t-n),u(t-1),\cdots,u(t-m),v(t)) \tag{5.6.46}$$

5.6.4 Other nonlinear control with fixed parameters

In this section other control methods for nonlinear systems are introduced.

Sliding mode

We have discussed feed-back linearization and nonlinear one-step-ahead control (using neural networks). The control law is designed, based on a nominal nonlinear model of the plant. We may obtain an exact model for the nominal plant by a neural network, and the controller can cancel the nonlinearity of the nominal plant. However, it is not clear how the control behaves in the presence of model uncertainties. From this point of view, it is crucial to consider robust nonlinear control. As a method of robust nonlinear control, sliding control has attracted interest recently. We can find the discussion of robust control using sliding mode methods in the book by Slotine et al. (1991).

Gain Scheduling

Gain scheduling is an approach to apply the well-developed linear control methods to the control of nonlinear systems. The idea is to select a number of operating points which cover the range of the system operation. At each operating point a linear controller is designed, based on an approximated linear model. Between operating points, the parameters of the controllers are then interpolated.

Fuzzy control

We sometimes encounter the situation where we have no way to acquire a model for a plant due to the complexity and highly nonlinear nature. However human-beings can sometimes control such badly-posed plant by the rules acquired empirically. The idea of fuzzy control is to directly acquire the linguistic rules from the human-beings and embed them in the control system. It is sometimes regarded as a sort of nonlinear version of gain scheduling. There is no theory for the stability and the performance of the control. We can find the concise discussion of fuzzy control in the literature by Yamakawa (1993).

5.7 Adaptive control

In this section we discuss adaptive control scheme. The basic approach treated in this thesis is to combine the parameter estimation technique shown in Section 5.4.1 with the control technique shown in Section 5.6.2.

Many approaches to adaptive control have been proposed. In particular two schemes have attracted interest: model reference adaptive control (MRAC) and self-tuning regulator (STR). Practical and theoretical discussions about MRAC can be found in Narendra and Annaswamy (1989). For STR we can refer to the book by Åstrom (1989). We can also find general views of adaptive control scheme in these books. Several treatments of adaptive controls for discrete-time linear systems were discussed by Goodwin and Sin (1984). The discussion in this section is adopted from the book by him.

MRAC was originally proposed to solve a problem in which the specifications are given in terms of a reference model which specifies how the plant should respond to the command signal. A block diagram of MRAC system is shown in Fig. 5.7.1. In this configuration the reference model is situated in parallel with the plant. The controller has two loops: an inner loop and an outer loop. The inner loop is a feed-back loop which generates a control signal, based on the control law. The parameters in the control law are adjusted by the outer loop in such a way that the error between the process output and the model output becomes small.

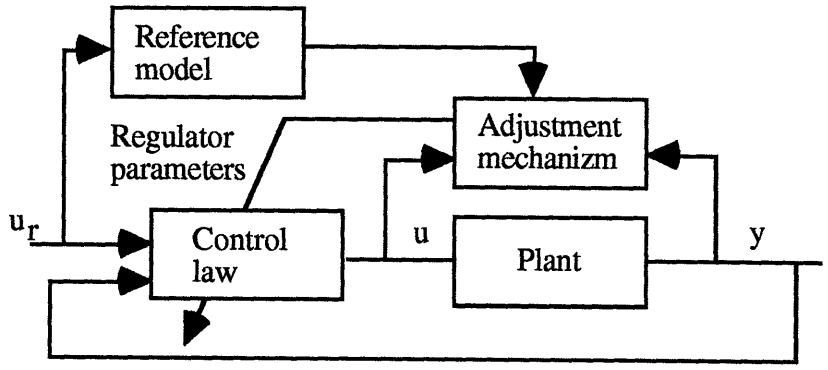


Fig. 5.7.1 Block diagram of MRAC

STR was originally proposed to deal with a stochastic regulation problem by an adaptive control scheme. The configuration of STR is shown in Fig. 5.7.2. It also has two loops, an inner loop, which is a feed-back regulator, and an outer loop, which is composed of an estimator of the parameters of the plant and an adjustment of the

parameters in control law. This configuration can cope with not only a stochastic regulation problem, but also a deterministic servo problem.

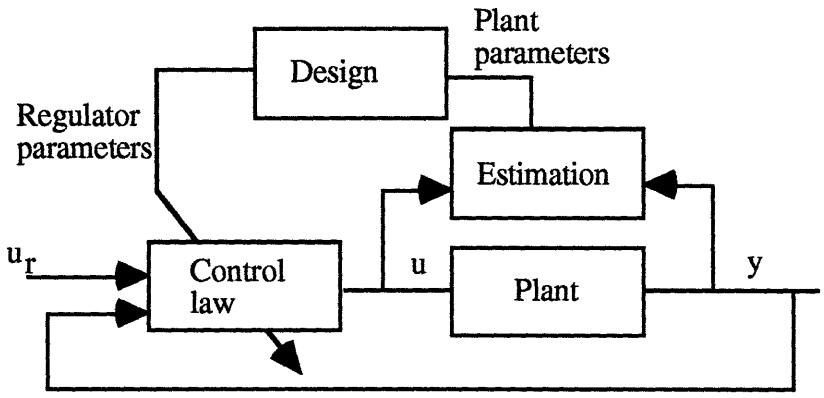


Fig. 5.7.2 Block diagram of STR

We sometimes use the terms of "direct" adaptive control and "indirect" adaptive control. In the direct adaptive control the controller parameters are updated directly. If the controller parameters are obtained indirectly via an estimation of the plant parameters we call it an indirect controller. The concept of MRAC scheme and STR scheme are very flexible with respect to the choice of adaptation algorithms. If a MRAC system involves an estimation of the plant parameters, it is indirect MRAC. If not, it is direct MRAC. Similar things can be said for STR.

One-step-ahead adaptive control, which was first presented by Goodwin, is a combination of a one-step-ahead predictive control law with an estimation law. The basic idea is almost identical to original STR scheme for deterministic servo problems. As discussed by Goodwin, various algorithms are possible. If we add a reference model, it becomes model reference one-step-ahead adaptive control. We can also implement both indirect and direct algorithms. The name "one-step-ahead" is not commonly used, but it will be called this way in this thesis. Since almost all one-step-ahead controllers that are discussed in this thesis are direct STR, the terms, "direct" and "STR" will not be used.

As we have seen, neural networks can have a framework which iteratively acquires a nonlinear property by an incremental identification method such as back-propagation method. This feature, similar to linear adaptive scheme, has recently attracted much interest, and brings about many approaches to nonlinear adaptive control schemes. In this section we shall also discuss the scheme of nonlinear adaptive controls using neural networks.

5.7.1 Linear adaptive control

One-step-ahead adaptive control

The idea of one-step-ahead adaptive control is simply to combine a parameter estimation technique with a one-step-ahead predictive control law.

Let us consider a controller composed of an estimation based on a projection algorithm, and a prototype one-step-ahead predictive control law.
The plant model is given by ARMA model, which leads to the predictor form:

$$\begin{aligned} &y(t+d) = \phi(t)^t \vartheta_0 \\ &\text{where} \\ &\phi(t)^t = (y(t),\cdots,y(t-n+1),u(t),\cdots,u(t-m-d-1)) \\ &\vartheta_0{}^t = (\alpha_0,\cdots,\alpha_{n-1},\beta_0,\cdots,\beta_{m+d-1}) \end{aligned} \tag{5.7.1}$$

Using a projection algorithm, the estimation law is given by

$$\begin{aligned} &\hat{\vartheta}(t) = \hat{\vartheta}(t-1) + M(t-1)\phi(t-d)e(t) \\ &M(t-1) = \frac{a(t)}{c + \phi(t-d)^t \phi(t-d)} \\ &e(t) = y(t) - \phi(t-d)^t \hat{\vartheta}(t-1) \end{aligned} \tag{5.7.2}$$

Then, the input is determined so that it satisfies $\phi(t)^t \hat{\vartheta}(t) = y_d(t+d)$
which results in the control law:

$$\begin{aligned} u(t) = \frac{1}{\hat{\vartheta}_{n+1}(t)}[&y_d(t+d) - \hat{\vartheta}_1(t)y(t)\cdots - \hat{\vartheta}_n(t)y(t-n+1) \\ &-\hat{\vartheta}_{n+2}(t)u(t-1)\cdots - \hat{\vartheta}_{n+m+d}(t)y(t-m-d+1)] \end{aligned} \tag{5.7.3}$$

The equations (5.7.2) and (5.7.3) are the adaptive control law. Since we start with the predictor form, it does not involve the estimation of the plant model. Therefore the approach is considered a sort of direct adaptive controller. It is noted that the algorithm requires a desired trajectory at any time. It is also remarked that the adaptive control leads to a time-varying nonlinear problem.

We may raise some questions for this control, such as convergence to the desired trajectory, and the boundness of the input and the output. However, due to the following theorem, the global convergence and the boundness are guaranteed under some assumptions.

Theorem 5.7.1 (Goodwin)
Subject to the following assumption:

(1) The time-delay d is known.
(2) An upper bound of the orders of the polynomials describing the system is known.
(3) The "inverse" model of the system is stable.

the adaptive control law given by (5.7.2) and (5.7.3) yields

(1) The inputs and the outputs are bounded.
(2) The output of the closed loop system converges to the desired trajectory.

This theorem indicates if we know the structure, or the time-delay and the order of the plant, and the system is a minimum phase in a sense, the adaptive control law above guarantees the most fundamental performance of the closed system, convergence and stability.

It is possible to combine the projection algorithm with any control laws which are discussed in Section 5.6.2. we can also use the least square algorithm. Any combination guarantees convergence and stability under the conditions similar to those in the above theorem. We can find detailed discussions in Goodwin and Sin (1984).

Model reference one-step-ahead adaptive control

The methods discussed above are direct STRs. We can also add the concept of MRAC to this scheme. The idea is to further specify the desired output $y_d(t)$ such that it is the output of a linear dynamic system driven by a reference input $r(t)$.

Suppose the linear dynamic system, or the reference model, is given by

$$E(q^{-1})y_d(t) = q^{-d}H(q^{-1})r(t) \qquad (5.7.4)$$

where the reference model should be stable. The objective of this control is to make $y(t) = y_d(t)$ without directly calculating the value of $y_d(t)$.

First we consider the modification of the predictor form, based on the fact we can obtain more general predictor form:

$$E(q^{-1})y(t+d) = \alpha(q^{-1})y(t) + \beta(q^{-1})u(t) \qquad (5.7.5)$$

where $\alpha(q^{-1})$ and $\beta(q^{-1})$ can be uniquely determined. (Goodwin)
Since we want to make $y(t) = y_d(t)$, (5.7.4) and (5.7.5) yield

$$\alpha(q^{-1})y(t) + \beta(q^{-1})u(t) = q^{-d}H(q^{-1})r(t) \qquad (5.7.6)$$

That is the control law of the model reference control. Obviously we need estimate $\alpha(q^{-1})$ and $\beta(q^{-1})$ on-line to achieve the adaptive scheme.

5.7.2 Nonlinear adaptive control

In this section we introduce nonlinear adaptive controls using neural networks centered on Narendra's paper (1990). As we mentioned in Section 5.3, neural networks identified by an incremental minimization, back-propagation, has a configuration suitable for an adaptive control scheme. Many researchers have raised much interest in this configuration and have proposed some approaches.

Model reference adaptive control using neural networks

Narendra and other researchers presented two types of control systems using neural networks. The one is MRAC type control system. Figure 5.4.4 shows the configuration of this control. Basic structure is as same as linear MRAC. The concept of this control is as follows.

1. The following reference model is predetermined a priori.

$$y_r(t+1) = f(y_r(t),\cdots y_r(t-l_1+1), r(t),\cdots, r(t-l_2+1)) \qquad (5.7.7)$$

where function $f(\cdot)$ is a nonlinear or liner stable model, and $r(k)$ is a reference input.

2. The neural network describing the plant $NN_p[\cdot]$ such that

$$\hat{y}(t+1) = NN_p[y(t),\cdots, y(t-n+1), u(t),\cdots, u(t-m+1)] \qquad (5.7.8)$$

is identified by an incremental identification method on-line so that it minimizes the cost function:

$$J(t) = \frac{1}{2}(y(t) - \hat{y}(t))^2 \qquad (5.7.9)$$

3. The neural network controller $NN_c[\cdot]$ such that

$$u(t) = NN_c[y(t),\cdots, y(t-i+1), u(t-1),\cdots, u(t-j+1), y_r(t+1)] \qquad (5.7.10)$$

is determined on-line so that it minimizes the cost function:

$$J(t) = \frac{1}{2}(y(t) - y_r(t))^2 \qquad (5.7.11)$$

When we use an incremental identification method, we need the derivatives of the actual plant system. Since they are not available, we use the derivatives of the plant model $NN_p[\cdot]$ instead of those of the actual plant.

4. The control signal in (5.7.10) is fed into the plant.

If instead we use Model 2 structure in Section 5.3 for the plant model, we don't need such a neural network as $NN_c[\cdot]$. We can design the controller by a linear manipulation. For the linear MRAC, the global stability is ensured. However, it is not clear whether the nonlinear MRAC using neural networks yields a stable result. Furthermore the convergence is not also guaranteed.

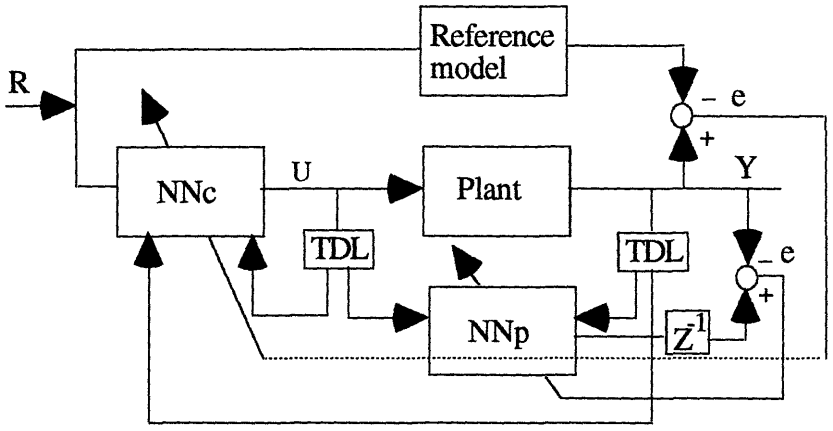


Fig. 5.7.3 Nonlinear MRAC using neural network

Self-tuning regulator using neural networks

We can also take STR configuration by similar discussions. (Chen (1990)) The plant model is identified in a way same as (5.7.8). If we make the control so that it minimizes the cost function:

$$J(t)=\frac{1}{2}(y(t+1)-y_d(t+1))^2 \qquad y_d(t+1)\text{: a desired trajectory} \quad (5.7.12)$$

then, the control results in a nonlinear self tuning regulator. Figure 5.7.4. shows the configuration of nonlinear STR. Obviously the configuration is an on-line version of nonlinear one-step-ahead control discussed in Section 5.6.2.

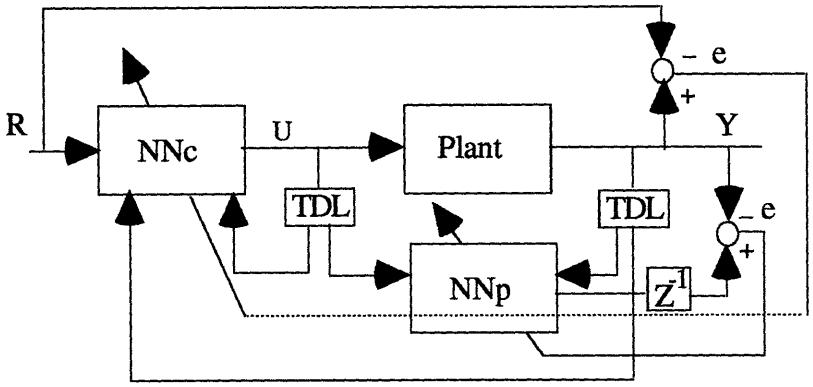


Fig. 5.7.4 Nonlinear STR using neural network

5.8 Control for cooling rate

We have so far discussed some identification and control methods. In this section we would like to consider what kind of control is applicable to the cooling rate control. The objective of the design is to get better performance than that of PI control shown in Chapter 4.

First let us consider a linear controller. In this case we usually use a linearized model as the nominal plant model around the operating point of interest. Two types of control schemes, feed-back with fixed parameters and adaptive control scheme are applicable.

When we design a feed-back controller with fixed parameters, we must focus on the robustness. We must design the control taking into consideration the whole discrepancy between the linear model and the actual nonlinear system.

Then, a question arises, how can we design the control? We can not precisely determine how much the actual plant is perturbed from the linear nominal model by its

nonlinearity. We could not, therefore, use robust control techniques for linear systems. Eventually, we have no way other than to resort to the empirical design. That is essentially equivalent to PI control design shown in Chapter 4.

From the viewpoint above the adaptive scheme is quite attractive, because the control can accommodate the perturbation automatically. Therefore, we can say one of feasible methods to improve the performance of the cooling rate control is linear adaptive control. However, some problems still remain. First can the estimator identify the parameters as fast as they change due to the nonlinearity? Second, does the linear adaptive controller show a stable behavior for our problems? The theorem concerning stability and convergence in adaptive scheme gives us an advantage. However it is applicable to a slow time-varying system. Therefore it does not necessarily guarantee the stability for our problem.

Next we will pay attention to a nonlinear controller. We have so far discussed controllers using neural networks. It is quite interesting to see how the nonlinear control works for our problem. Therefore, as the second trial, we would like to consider a controller using neural networks.

In Chapter 6, we will discuss the performance of a linear adaptive control. We shall see how the control using neural networks work in Chapter 7.

Chapter 6

Application of linear adaptive control to cooling rate

The scheme of linear adaptive control was reviewed in Section 5.7.1. In this Chapter we discuss the application of the adaptive scheme to the cooling rate. We also show the performance by simulating the resulting controlled system.

We will use one-step-ahead adaptive control as the method, since the procedure of the design is well established, as shown in Section 5.7.1. First we have to choose the control law and the estimation law. Second, we have to consider some practical modifications in accordance with the nature of the process. The objective of this chapter is to show how we can design the one-step-ahead adaptive control for the cooling rate.

Some considerations on the choice of the adaptive controller are discussed in Section 6.1. Section 6.2 copes with the problems that take place in practice. Two types of model structures are proposed to deal with a varying time-delay in Section 6.3. The simulation results are shown to evaluate the performance of the adaptive controller in Section 6.4. The final comments are made in Section 6.5.

6.1 Choice of adaptive controller

The design of an adaptive controller requires an appropriate choice of the estimation algorithm and the control law. We must also choose the model structure and the sampling interval. There are no precise rules for the choice. Basically it depends on the nature of the system and the objectives of the control. Therefore, it is quite important to observe the behavior of the system, and have a physical insight into it.

When we hold the discussion about the choice, we must keep in mind the following features of the cooling rate.

1. The objective of the control is to regulate the cooling rate. Therefore, our concern lies in the disturbance rejection and the noise suppression. In particular, we are concerned with how fast the controller gets rid of the disturbances, compared with the conventional controller, PI control.
2. Goodwin et al (1984) proved one-step-ahead adaptive control is always stable for a time-invariant linear system under some conditions as discussed in Section

5.7.1. He also suggests the adaptive control is applicable to a "slow time-varying system".

However, we will use the adaptive control as a system to adjust itself to the nonlinear dynamics of the plant. Therefore, the discussion made by Goodwin et al for the stability does not necessarily agree with our objective. In this thesis we just show how the adaptive control performs for this problem. More theoretical considerations should be discussed.

3. As the problem associated with **2,** the dependence of the time-delay on the input and the state of the system completely violates the conditions that guarantee the stability of the control.

4. The observations of the cooling rate response by the travel speed show that it behaves like a first order system. Therefore, it is quite reasonable to approximate its dynamics as a first order system.

We can also use a higher order model, however, it should be an overdamped model since the cooling rate dynamics shows such response. A bigger relative degree of the transfer function gives us a gentler slope of the rising in the initial response by the step input. We use a higher order model simply to accommodate such a gentler slope of the response which cannot be described by a first order model.

5. The response of the cooling rate never shows an undershoot behavior. Therefore the model should be minimum phase. (It is obvious that the system is stable.)

6. Unmodeled dynamics, particularly in a higher frequency, sometimes brings about a big constraint on the design of a controller. However, as for the cooling rate control, we would not have to be worried about this problem, considering the characteristics of the heat transfer. We do not have to suspect its dynamics has a resonance in a high frequency.

In what follows we shall discuss the choice of the control law, the estimation algorithm, and the sampling period, considering the above features of the system. We shall also discuss the model structure in Section 6.3.

Choice of control law

Of the control laws; prototype one-step-ahead adaptive control, weighted one-step-ahead adaptive control, model reference adaptive control, and pole assignment adaptive control, discussed in Section 5.7.1, a model reference adaptive controller is typically used for a tracking problem, and is not of interest here.

The other three control schemes are applicable to a regulation problem. A one-step-ahead controller determines the input so that the output is exactly same as a desired output. Therefore, there is a strong possibility that the controller will excite undesirable dynamics in a high frequency with the presence of a model error, but we are not concerned with the problem as we discussed above. However, we can not forget the existence of the time-delay. The estimation error would also yield bad performance with the presence of the time-delay.

The simplest way to avoid such a bad performance is a weighted one-step-ahead controller. The weighted one-step-ahead controller simply reduces the size of the input, which results in the reduction of an unfavorable response.

We can also use a pole assignment controller. However, the plant is a nonlinear system which sometimes shows a first-order-system-like response, and sometimes shows a second-order-system-like response. Suppose the actual system is of second order and the model structure is of first order. Then it is impossible to say how the system behaves for the assigned poles. Therefore, the significance of the pole assignment method will be lost. Furthermore the implementation is more complicated than that of the weighted one-step-ahead control.

From the points discussed above, we will use the weighted one-step-ahead controller:

$$u(t) = \frac{\beta_0 \left\{ y_d(t+d) - \alpha(q^{-1}) y(t) - \beta'(q^{-1}) u(t) \right\}}{\beta_0^2 + \lambda} \tag{6.1.1}$$

as the control law. (See Section 5.6.2) It is known that this control yields a steady-state error. Therefore, we may need a modification for the control if the error is unfavorably large. There is not a theory for adjusting the weight. Therefore, we have to determine it by trial and error until we get satisfactory performance.

Choice of parameter estimation algorithm

Recursive least square estimation algorithm is much superior to the projection algorithm in terms of the convergence rate. Therefore we will use the least square algorithm because the performance of the adaptive control depends strongly on how fast the estimation keeps up with the change of the parameters caused by the nonlinearity of the system.

The fundamental drawback of this algorithm is that the gain matrix diminishes to zeros with time. Once the gain matrix reaches zeros, nothing happens in the estimation. Therefore we need to modify the algorithm to cope with this problem.

Selection of sampling period

It is recommended that the sampling period be roughly less than one fifth of the fastest time constant of interest. The minimum sampling period available for measuring the cooling rate is 0.5 sec due to the computational limitation of the computer. Roughly the fastest time constant, dependent on the input, is about 3 sec. Therefore, the recommended condition is satisfied if we use the minimum sampling period of 0.5 sec.

However, the small sampling period would have some drawbacks. First, it increases the number of parameters that should be estimated because of the presence of the time-delay. (See the discussion in Section 5.6.2.) Second the parameter would be rather sensitive to the noise. (In a sense a sampling has an effect of a filtering.)

Basically, the selection of the suitable sampling period is a very hard problem, because a minimum phase continuous-time system may transform to a discrete-time system with unstable zeros (non-minimum phase discrete-time system) due to a bad selection of the sampling period. If the zeros are unstable, one-step-ahead control law may yield an unbounded input signal to the plant.

The step response of the cooling rate shows that its dynamics should be minimum phase. Therefore, there should exist a suitable sampling period. However, we have no clues for the selection of the sampling time due to the nonlinear nature of the system.

Considering the time-delay and time constant of the plant and the discussion above, we will take 1 sec for the sampling time although we can not say whether it is actually suitable.

6.2 Practical aspects of controller design

6.2.1 Modifications of least square algorithm

Various modifications of the recursive least square estimation have been proposed to tackle the problem that the gain matrix diminishes with time. (Goodwin (1984), Landau (1990), Ljung (1987)) The choice of the method basically depends on the nature of the system and the disturbance and noise imposed on the system. The simplest method among them is forgetting factor method.

Forgetting factor method

This method proposes the recursive algorithm:

$$\hat{\vartheta}(t) = \hat{\vartheta}(t-1) + M(t-1)\phi(t-d)e(t)$$

$$M(t-1) = \frac{P(t-2)}{\eta(t-1) + \phi(t-d)^t P(t-2)\phi(t-d)}$$

$$\mathrm{P(t-1)} = \frac{1}{\eta(t-1)}\left[\mathrm{P(t-2)} - \frac{\mathrm{P(t-2)}\phi(t-d)^t\phi(t-d)\mathrm{P(t-2)}}{\eta(t-1) + \phi(t-d)^t P(t-2)\phi(t-d)}\right] \qquad (6.2.1)$$

where $\eta(t) = \eta_0 \qquad 0 < \eta_0 < 1$

This iteration results from the minimization of the cost function:

$$J(t) = \sum_{i=1}^{t} \eta_0^{t-i}[y(i) - \hat{\vartheta}(t)\phi(i-d)]^2 \qquad (6.2.2)$$

η_0 is known as a forgetting factor. It is seen that it has the effect of discarding the old data.

The constant forgetting factor method has another problem. If the data is not persistently exciting, the gain matrix P grows exponentially. To avoid this, Fortescue et al. (1981) proposed the following algorithm:

$$\eta(t) = 1 - \eta' \frac{e(t)^2}{1 + \phi(t-d)^t P(t-2)\phi(t-d)} \qquad (6.2.3)$$

where $e(t)$ is a prediction error.

When $\eta(t) = 1$ the estimation algorithm is essentially equivalent to the prototype least square algorithm. Therefore the estimation is turned off when the prediction error is equal to zero. On the other hand, the presence of the error continues the estimation. Since the average prediction error is proportional to the term $1 + \phi(t-d)^t P(t-2)\phi(t-d)$, the size of the error is normalized by dividing the error by the term $1 + \phi(t-d)^t P(t-2)\phi(t-d)$ in (6.2.3).

In the simulation in Section 6.4 the constant forgetting factor method is used since we have not seen the behavior of an exponential growth in the gain matrix.

6.2.2 Constrained parameter estimation

Goodwin et al. recommend some allowable range be given to the parameter estimates if we know the boundary region. The method to cope with the boundary is called constrained parameter estimation. The concept of the constrained parameter estimation is quite simple. It just selects the point in the region surface closer to the estimates made by the conventional algorithm discussed above when the estimate is located outside the region. However, the actual implementation is quite complicated for an arbitrarily prescribed region.

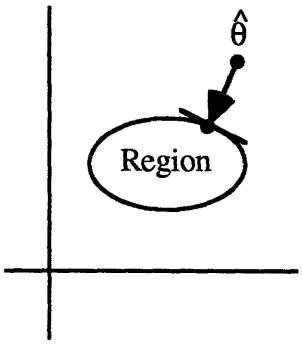


Fig. 6.2.1 Constrained parameter estimation

Consider the one-step-ahead control law. It simply drives the input so that the future output can follow a desired trajectory. (See the equation (6.1.1).) We can easily see the sign of the parameter β_0 plays a key role to determine whether the input should be increased or decreased to achieve the desired value. The controller would fail if the estimate for β_0 has a wrong sign. Therefore we may need to constrain at least the parameter β_0.

If we constrain only the parameter β_0, the algorithm for the constrained estimation can be reduced to the following simple algorithm.

$$\beta_0 = \begin{cases} \beta_u & (\text{if } \hat{\beta}_0 > \beta_u) \\ \beta_l & (\text{if } \hat{\beta}_0 < \beta_l) \\ \hat{\beta}_0 & (\text{Otherwise}) \end{cases} \qquad (6.2.4)$$

where β_l is the lower bound, β_u is the upper bound and $\hat{\beta}_0$ is the estimate by the conventional estimation algorithm.

In Section 6.4 we shall see how important the boundary limit of the parameter β_0 is for the adaptive control.

6.2.3 Weighted one-step-ahead control with an integrator

The weighted one-step-ahead controller yields a steady-state error as mentioned in Section 6.1. It is a "common sense" of control engineers to put an integrator in the control so that we can reduce the steady-state error. We would like to discuss how we can introduce an integrator into the weighted one-step-ahead-controller.

Introduce a new variable $\bar{u}(t)$ such as

$$\bar{u}(z) = \frac{1-z^{-1}}{1-\gamma z^{-1}} u(z) \qquad 0 < \gamma < 1 \tag{6.2.5}$$

and consider the cost function $J(t)$ such as

$$J(t) = \{\frac{1}{2}(y(t+d) - y_d(t+d))^2 + \lambda \frac{1}{2}\bar{u}(t)^2\}. \tag{6.2.6}$$

The system (6.2.5) is stable. Therefore the value of $\bar{u}(t)$ converges to zero if $u(t)$ is bounded and satisfy

$$\lim_{t\to\infty} u(t) = u_\infty \qquad \text{where } u_\infty \text{ is a constant.}$$

Therefore if $\lim_{t\to\infty} y_d(t) = y_\infty$, then the output $y(t)$ becomes gradually closer and closer to y_∞ by the control law to minimize the cost function (6.2.6) because the value of $\bar{u}(t)$ gradually converges to zero. Thus the control described in (6.2.5) introduces integral action and eliminates the steady state error.

The control to minimize the cost function (6.2.6) is given by:

$$u(t) = \frac{\beta_0\{y_d(t+d) - \alpha(q^{-1})y(t) - \beta'(q^{-1})u(t)\} + \lambda\{u(t-1) - \gamma\bar{u}(t-1)\}}{\beta_0^2 + \lambda} \tag{6.2.7}$$

If $\beta_0 \neq 0$ the relation between $y(t)$ and $y_d(t)$ is described by:

$$\{(1-\lambda q^{-1})B'(q^{-1}) + \frac{\lambda}{\beta_0}(1-q^{-1})A(q^{-1})\}y(t+d) = P(q^{-1})B'(q^{-1})y_d(t+d) \qquad (6.2.8)$$

We can easily verify the effect of integral action by (6.2.8).

If β_0 becomes zero at some time t_1 and remains zero after the time, the discussion above does not hold. Then the control law (6.2.7) produces such an input sequence as:

$$u(t) = u(t-1) - \gamma\bar{u}(t-1) \qquad (6.2.9)$$

The equations (6.2.5) and (6.2.9) imply $u(t)$ does not change after the time t_1. Therefore, the error caused by β_0 whose value is zero will remain as a steady-state error.

6.2.4 Scaling variables

It is recommended to scale variables so that they have the same order of magnitude. If the values of the output variable are much greater than those of the input variable, the elements of the matrices in the estimation algorithm may include quite large numbers and quite small numbers. It is well known the manipulation using such matrices sometimes yield a large numerical error. To avoid this numerical error we should scale the input variable and the output variable so that they have a same "size".

We use the following scaling in the simulation:

$$\begin{aligned} u &= v - 6.0 \\ y &= (cr - 60.0)/20.0 \end{aligned} \qquad (6.2.10)$$

where u is the scaled input variable, v is the travel speed, y is the scaled output variale, and cr is the cooling rate.

The numbers of 6.0 and 60.0 are selected, supposing the regulation around 60 °C/sec.

6.2.5 Modified one-step-ahead adaptive controller

We have so far discussed the modifications of one-step-ahead adaptive control to cope with the problems that take place in practice. Figure 6.2.2 shows the control law and estimation law in the modified one-step-ahead adaptive control.

As shown in Section 5.6.2, any ARMA model with a time-delay can be converted to d-step-ahead prediction form:

$$y(t+d) = \alpha(q^{-1})y(t) + \beta(q^{-1})u(t) = \phi(t)^t \vartheta \qquad (6.2.11)$$

where $\alpha(q^{-1}) = \alpha_0 + \alpha_1 q^{-1} + \cdots + \alpha_{n-1} q^{-(n-1)}$

$$\beta(q^{-1}) = \beta_0 + \beta_1 q^{-1} + \cdots + \beta_{n_1+d-1} q^{-(n_1+d-1)}$$

$$\phi(t) = [y(t), \cdots, y(t-n), u(t), \cdots, u(t-m)]$$

$$\vartheta = [\alpha_0, \cdots, \alpha_n, \beta_0, \cdots, \beta_m] \qquad \beta_0 \neq 0$$

Therefore the adaptive control is based on d-step-ahead prediction. Goodwin et al., however, call it "one-step-ahead" adaptive controller. Therefore we would like to keep their convention and call it in this way.

Estimation law:

(The recursive least square algorithm with a forgetting factor)

$$\hat{\vartheta}(t) = \hat{\vartheta}(t-1) + M(t-1)\phi(t-d)e(t)$$

$$M(t-1) = \frac{P(t-2)}{\eta(t-1) + \phi(t-d)^t P(t-2)\phi(t-d)} \qquad (6.2.12)$$

$$\mathrm{P(t-1)} = \frac{1}{\eta(t-1)}\left[\mathrm{P(t-2)} - \frac{\mathrm{P(t-2)}\phi(t-d)^t\phi(t-d)\mathrm{P(t-2)}}{\eta(t-1) + \phi(t-d)^t P(t-2)\phi(t-d)}\right]$$

Control law:

(The weighted one-step-ahead control with an integrator)

$$u(t) = \frac{\hat{\beta}_0\left\{y_d(t+d) - \hat{\alpha}(q^{-1})y(t) - \hat{\beta}'(q^{-1})u(t)\right\} + \lambda\{u(t-1) - \gamma\bar{u}(t-1)\}}{\hat{\beta}_0^2 + \lambda} \qquad (6.2.13)$$

where

$$\left(1-\gamma q^{-1}\right)\bar{u}(t) = \left(1-q^{-1}\right)u(t) \qquad 0 < \gamma < 1$$

$$\hat{\beta}'(q^{-1}) = \hat{\beta}'(q^{-1}) - \hat{\beta}_0$$

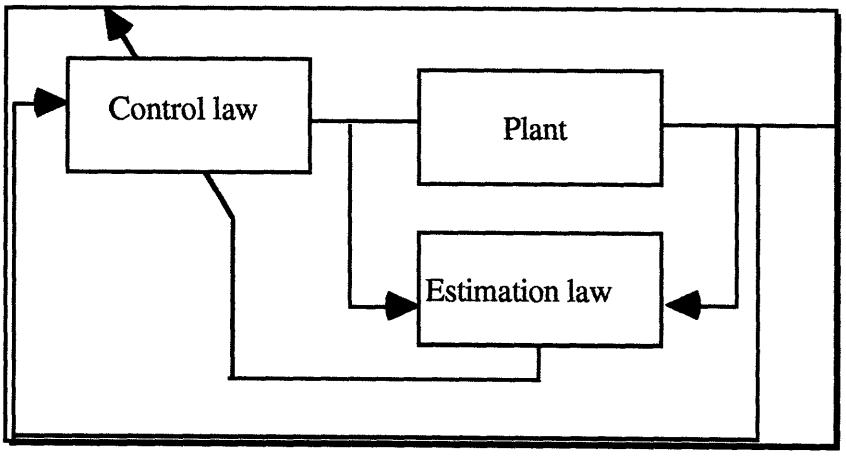


Fig. 6.2.2 Adaptive control

6.3 Model structure

The step response of the cooling rate is shown again in Fig. 6.3.1. Using the figure, we would like to discuss how we determine the model structure.
The observation of this figure shows

1. When we decreases the transfer speed, the cooling rate shows a tendency of a first order response after a time-delay.
2. However, the response shows a tendency of a second or higher order system with some relative degree when the travel speed increases.
3. The time-delay varies, depending on the input. The figure shows it varies from 2 sec to 6 sec.

The critical problem is the variation of the time-delay. In what follows two types of possible model structures are proposed so that we can accommodate the change of the time-delay. Note that the conditions under which the stability and the convergence to a desired output are guaranteed are violated. Therefore, both structures may yield an unfavorable behavior.

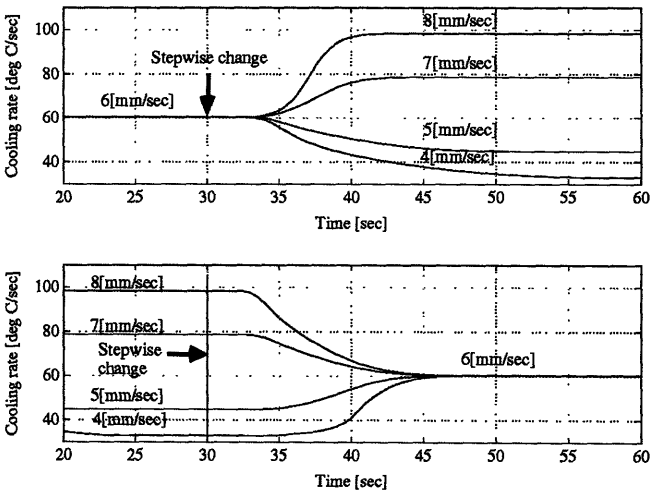


Fig. 6.3.1 Step response

6.3.1 Type 1: Approximation of time-delay

In this type the following assumptions are made.

1. The time-delay is assumed to be the minimum value, or 2 sec.
2. The delay-free part is assumed to take a third-order transfer function:

$$\frac{1-sT_d/2+(sT_d)^2/2}{1+sT_d/2+(sT_d)^2/2}\cdot\frac{k}{s/\tau+1} \tag{6.3.1}$$

so that we can accommodate the change of the time-delay.

The first fraction of the transfer function is simply Padé approximation for the time-delay, and the second fraction is the delay-free part of the plant dynamics, assuming the plant is a first-order system.

The assumptions above give us the ARMA model:

$$A(q^{-1})y(t) = B(q^{-1})u(t) \tag{6.3.2}$$

where

$$A(q^{-1}) = 1 + a_1 q^{-1} + a_2 q^{-2} + a_3 q^{-3}$$

$$B(q^{-1}) = q^{-3}(b_0 + b_1 q^{-1} + b_2 q^{-2}) \qquad b_0 \neq 0$$

since the z-transform of the transfer function would be written as

$$Z\left(\frac{1-e^{-s}}{s}\frac{1-sT_d/2+(sT_d)^2/2}{1+sT_d/2+(sT_d)^2/2}\frac{k}{s/\tau+1}\right) = z^{-1}\frac{(b_0+b_1z^{-1}+b_2z^{-2})}{1+a_1z^{-1}+a_2z^{-2}+a_3z^{-3}} \tag{6.3.3}$$

Therefore, its predictor form is given by:

$$y(t+3) = \theta^t \phi(t) \tag{6.3.4}$$

where

$$\theta^t = (\alpha_0, \cdots, \alpha_2, \beta_0, \cdots, \beta_4)$$

$$\phi(t)^t = (y(t), \cdots, y(t-2), u(t), \cdots, u(t-4))$$

We may need to give a lower bound to the estimate of β_0. We can easily understand the value of β_0 should be positive. However, we cannot determine exactly a lower bound other than zero because the model accommodates the change of the time-delay.

A possible problem is that the parameter β_0 has a quite small value, or essentially equal to zero. Then, the one-step ahead control law would yield a steady-state error even if the controller includes an integrator.

6.3.2 Type 2: Multi-model structure

In this type we give the following 5 model structures. In these models the structure of delay-free part is same, but they have different time-delay.

$$A_n(q^{-1})y(t) = B_n(q^{-1})u(t) \qquad (6.3.5)$$

where

$$A_n(q^{-1}) = 1 + a_{n1}q^{-1} + a_{n2}q^{-2}$$

$$B_n(q^{-1}) = q^{-(n+2)}(b_{n0} + b_{n1}q^{-1}) \quad b_0 \neq 0 \quad n = 1,\cdots,5$$

In this model structure, we assume the delay-free part of the system is a second order system.

Under the assumption above, the predictor form is given by:

$$y(t+d) = \theta_n{}^t \phi(t) \qquad (6.3.6)$$

where

$$d = n+2$$

$$\theta_n{}^t = (\alpha_{n0}, \alpha_{n1}, \beta_{n0}, \cdots, \beta_{nm}) \quad m = n+2$$

$$\phi(t)^t = (y(t), y(t-1), u(t), \cdots, u(t-n-2))$$

$$n = 1,\cdots,5$$

All the parameters of the models are estimated based on the least square algorithm. The estimation for a model is independent of that for another model. Therefore it requires 5 estimation cycles every sampling period. The model that provides the best prediction is used for the one-step-ahead control. The switch-over from one model to another may take place every control cycle.

One of the good points of this structure is that we can give a lower bound other than zero to the parameter β_{n0}. Thus we can avoid a steady-state error.

6.4 Performance of estimation and control

Some simulations are run to see how the one-step-ahead adaptive control performs for the cooling rate control. The same step input as PI control case shown in the Chapter 4 is used in these simulations unless the description of the reference input is mentioned.

6.4.1 Performance of Type 1 model structure

Here we discuss the performance of Type 1 model structure that is given by

$$y(t+3) = \theta^t \phi(t) \tag{6.4.1}$$

where

$$\theta^t = (\alpha_0, \cdots, \alpha_2, \beta_0, \cdots, \beta_4)$$

$$\phi(t)^t = (y(t), \cdots, y(t-2), u(t), \cdots, u(t-4))$$

(1) Estimation and prediction

The prediction results are shown in Figs. 6.4.1 and 6.4.2. We take 0.95 as the value of the forgetting factor. All initial values of the parameter estimates are set to zero. No boundaries are given to the parameter estimates.

In Fig. 6.4.1 the value of the travel speed is changed stepwisely every 30 sec. In Fig. 6.4.2 it is changed from 4 mm/sec to 6 mm/sec, based on a pseudo random sequence (PRBS). PRBS is widely used so that we can give a persistently exciting input to the plant. (See Landau (1990).) In this simulation PRBS is used simply to show how the estimation performs for the input with a rich spectral in frequency.

It is seen that the prediction follows the change of the output quite well. Even at the initial stage of the estimation, the almost same value as the actual one of the cooling rate is predicted.

The value of the estimated parameters changes drastically every time the input is changed. The changes would come from the nonlinear nature of the process. It is hard to evaluate the estimated parameters. Some unrealistic estimations are observed. The value of β_0 must be positive. However, it is sometimes estimated to be negative. Such estimation would yield a bad control.

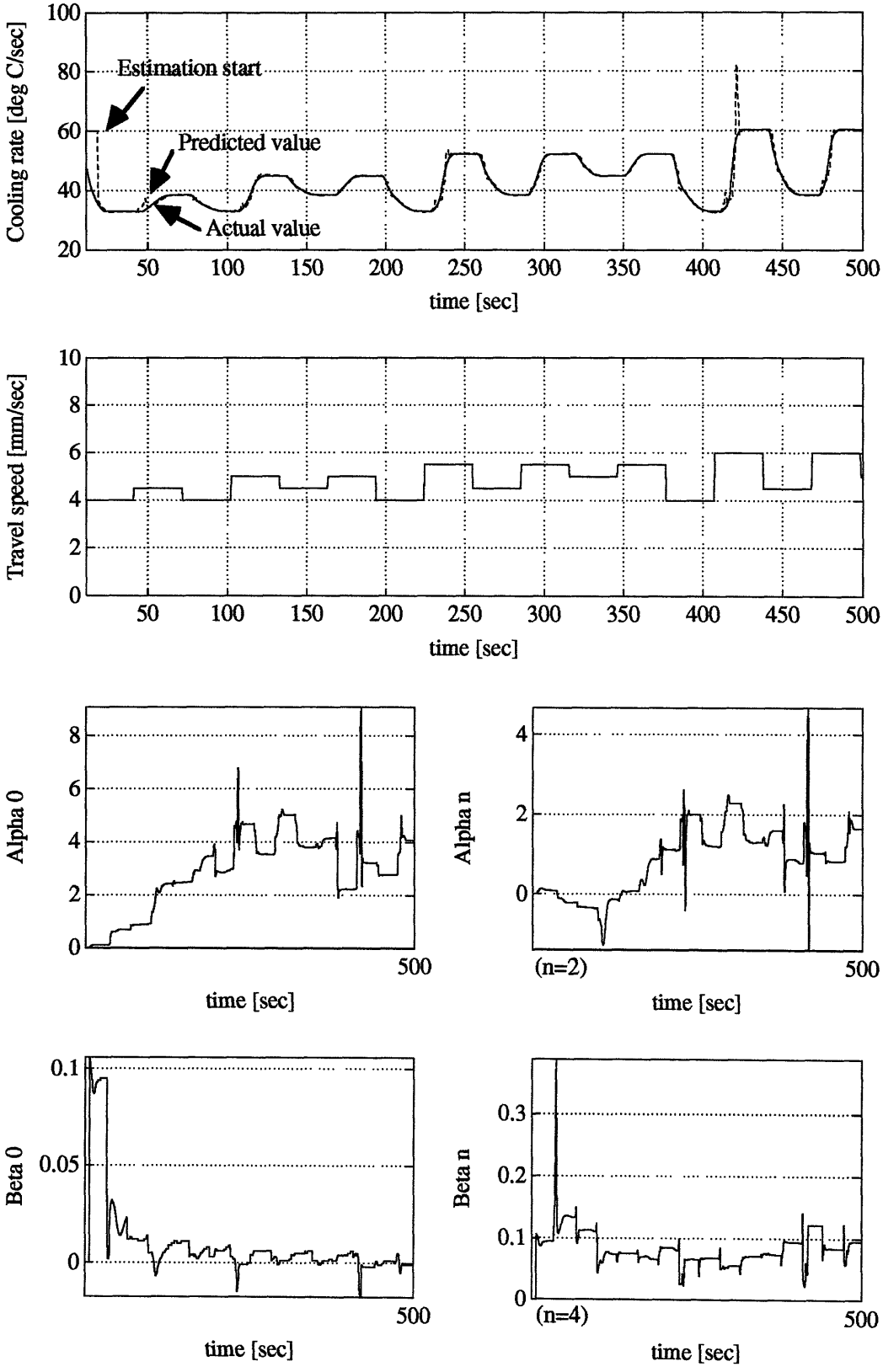


Fig. 6.4.1 The result of prediction for varied input magnitude

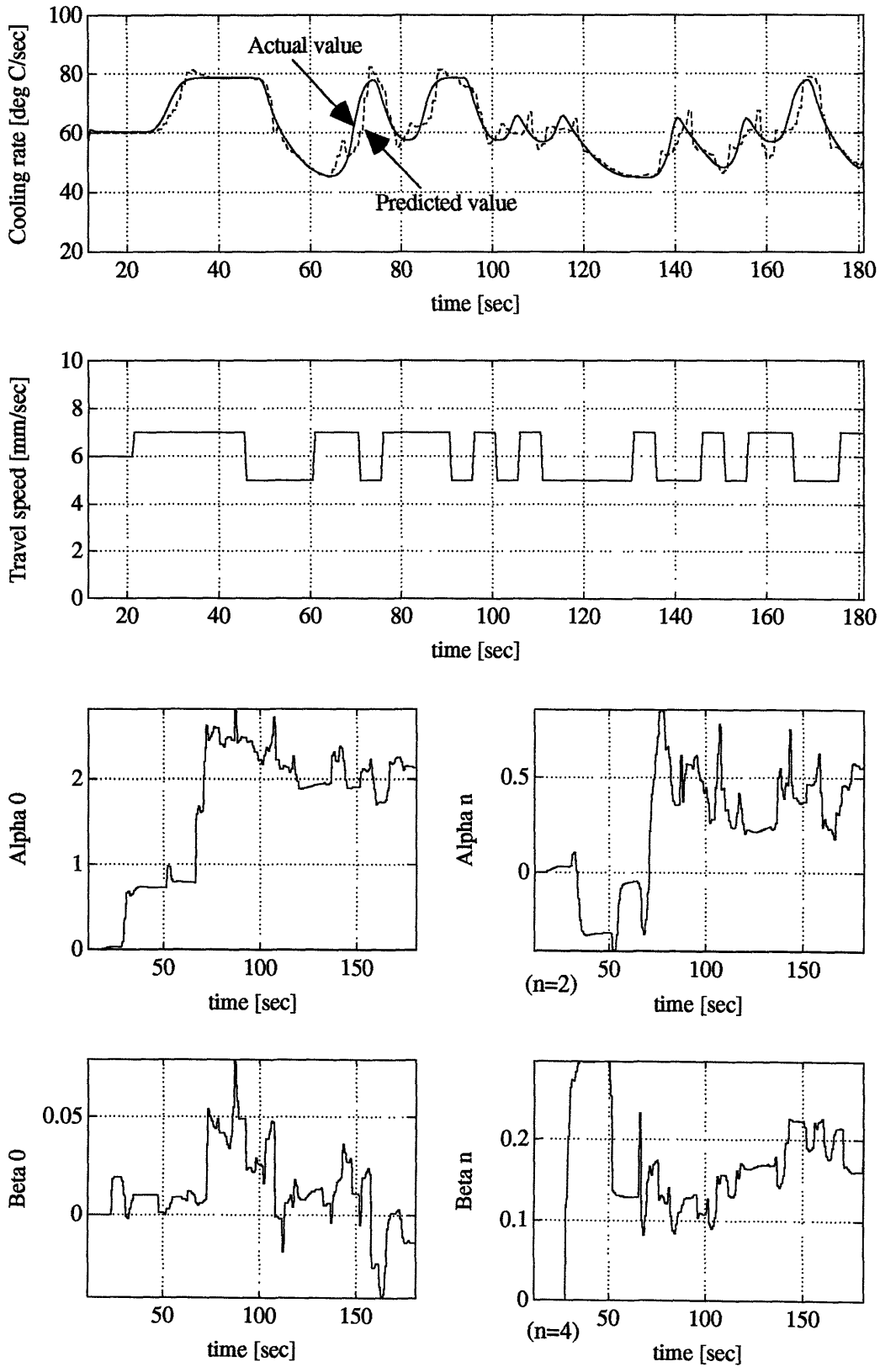


Fig. 6.4.2 The result of prediction for PRBS

(2) Adaptive control without integrator

Figure 6.4.3 through 6.4.5 show the results of the weighted one-step-ahead adaptive controller without an integrator. All initial values of the parameter estimates are set to zero. No boundaries are given to the parameter estimates. We take 0.05, 0.03, and 0.01 as the weight λ of the controller.

The response involves a large steady-state error. It comes from the nature of the weighted algorithms as discussed in Section 6.4.3. The steady-state error for the reference input of 80 °C/sec differs from that for 40 °C/sec. When the reference input is 40 °C/sec, the estimate of β_0 approaches zero. That is the reason it has a large steady-state error for the reference input of 40 °C/sec.

The decrease of the weight λ increases the slope of the response. Therefore we can achieve a faster response by using a small weight. However, quite small weight produces an instability by the control as shown in Fig. 6.4.5. The prototype one-step-ahead control is equivalent to the case that the weight λ is zero. Therefore the prototype one-step-ahead-control does not work for this problem.

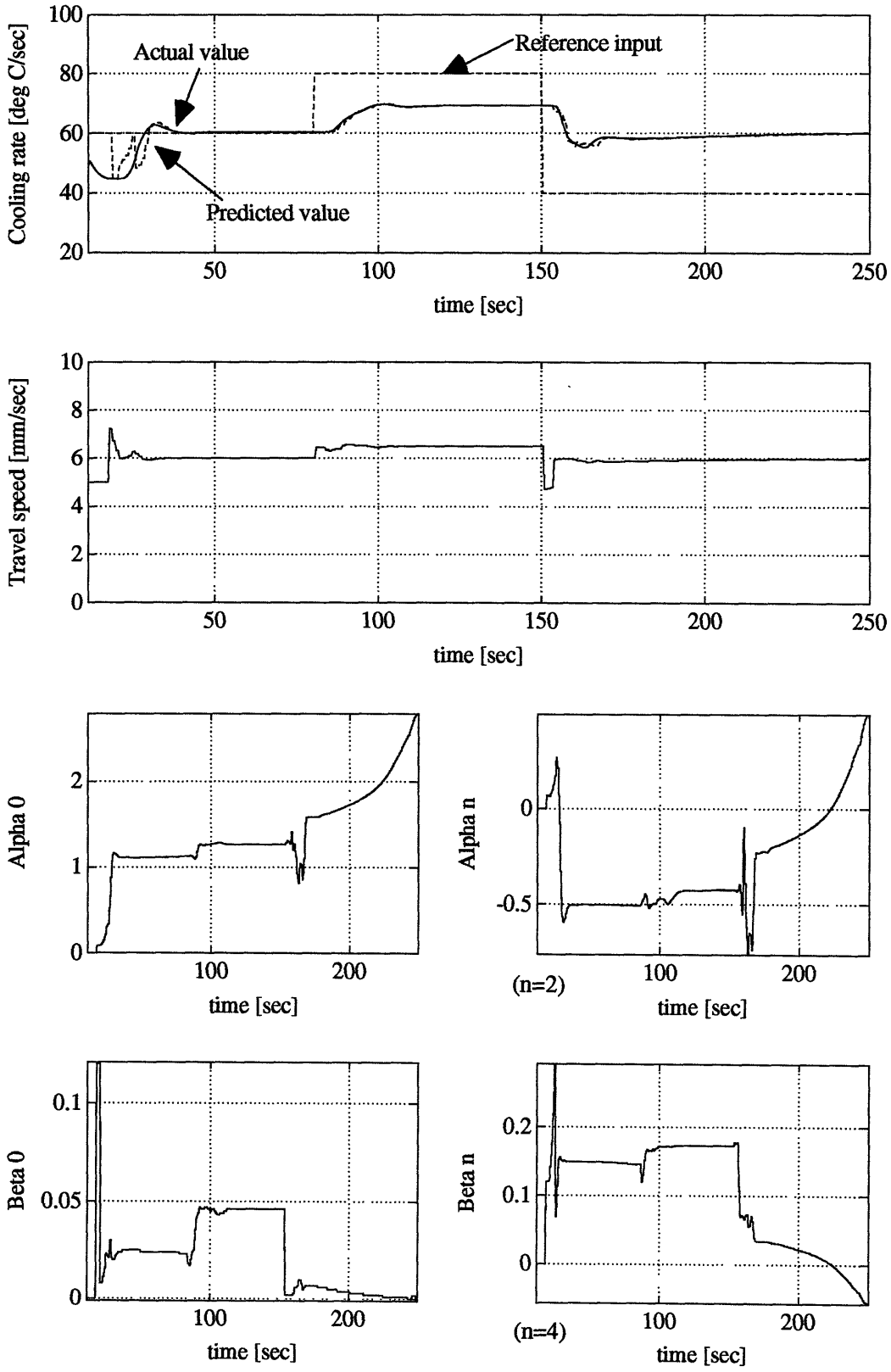


Fig. 6.4.3 One-step-ahead control without integrator (λ =0.05)

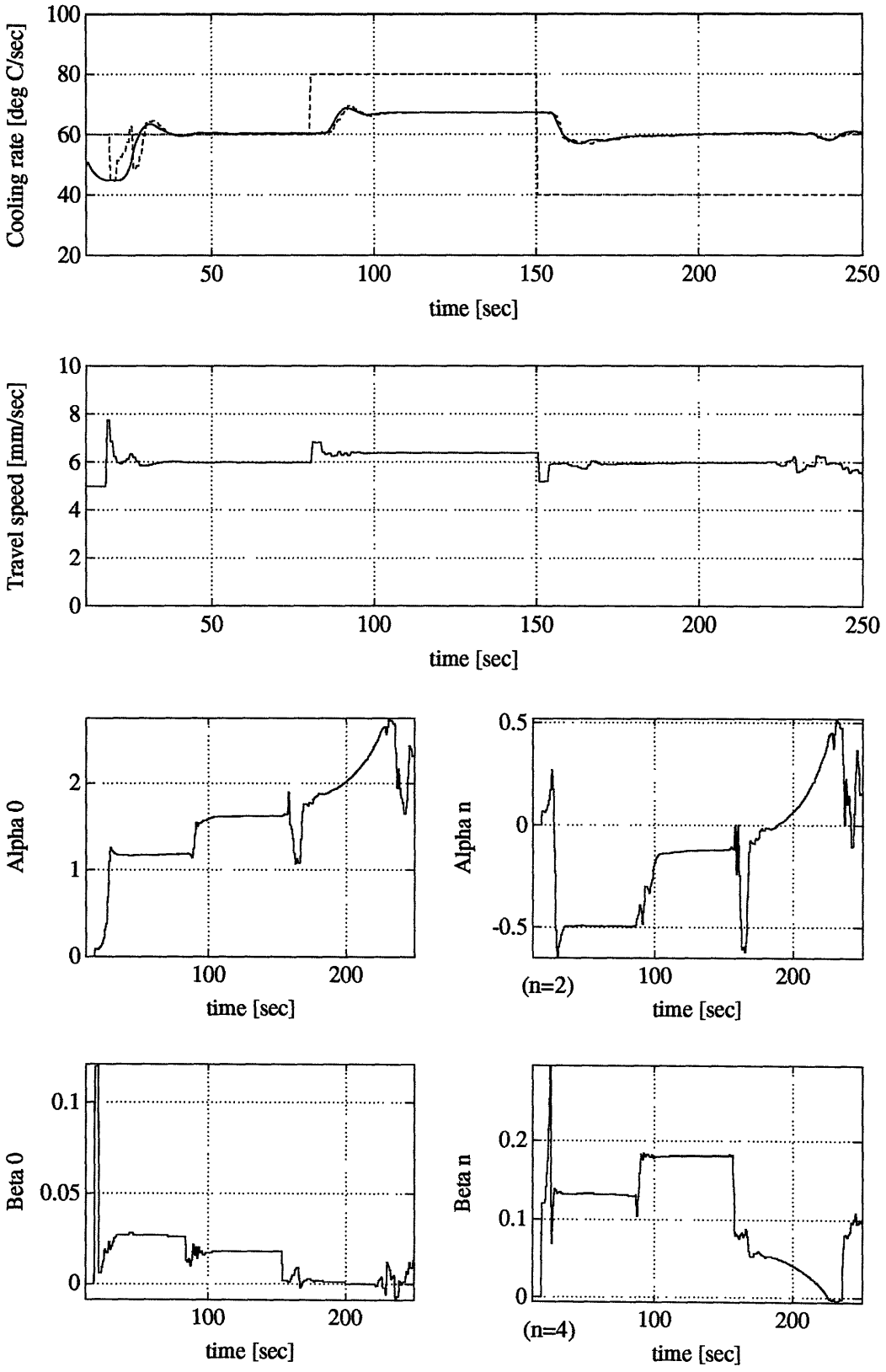


Fig. 6.4.4 One-step-ahead control without integrator (λ =0.03)

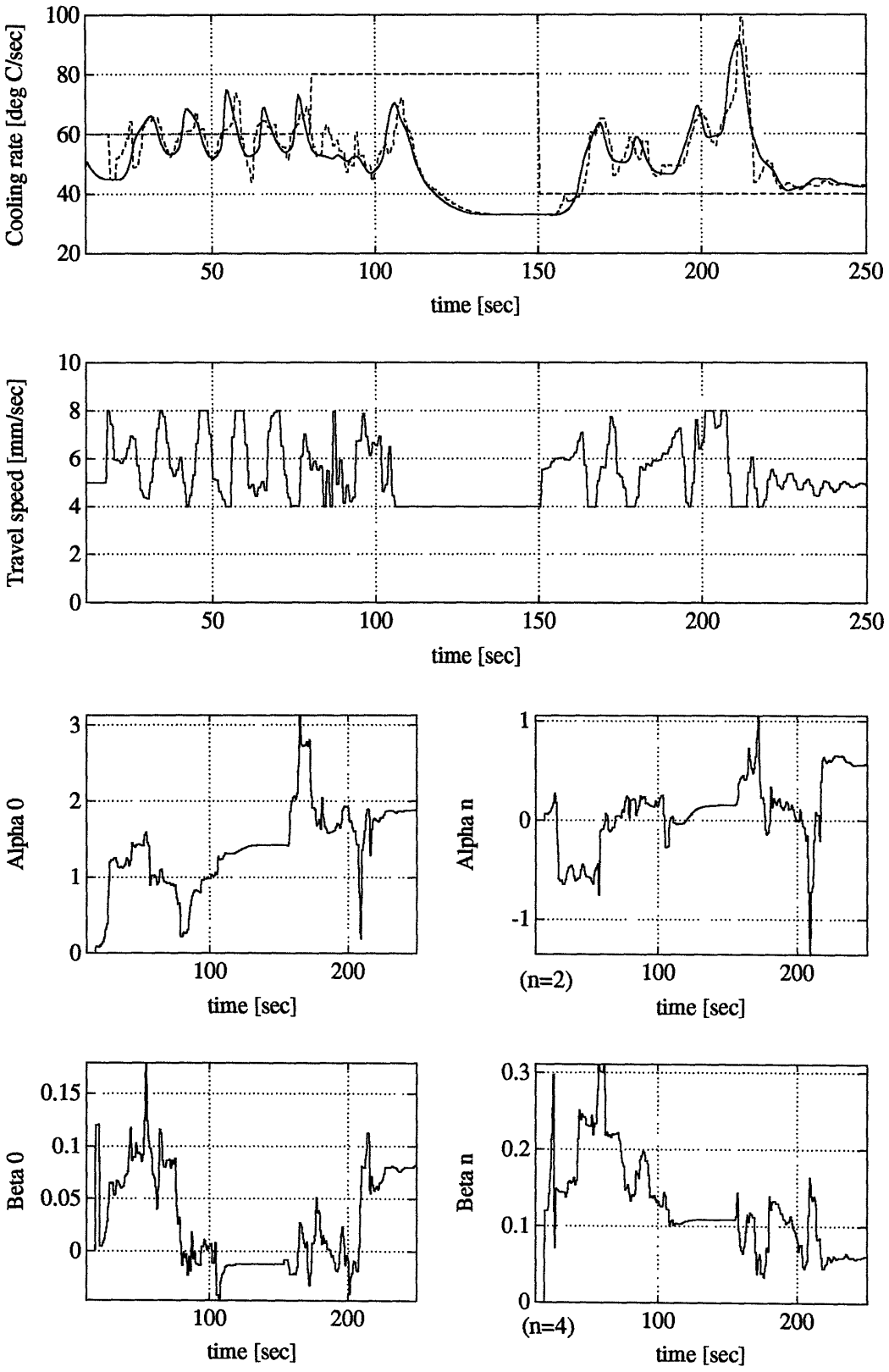


Fig. 6.4.5 One-step-ahead control without integrator (λ =0.01)

(3) Effect of integrator

An integrator is introduced to get rid of the steady-state error in Fig. 6.4.6 and 6.4.7. In order to see the importance of the lower limit of the parameter β_0, no lower limit is given for the case of Fig. 6.4.6 while the parameter estimate is bounded as β_0 is greater than zero in Fig. 6.4.7. In both figures we take 0.03 as the weight λ of the controller and 0.09 as the factor γ of the integrator.

In Fig. 6.4.6 the cooling rate first follows the reference inputs and reaches the value of 80 °C/sec and gradually decreases. When 40 °C/sec is given as the reference input, the cooling rate is driven to the wrong direction and saturates. Such bad performance takes place due to no lower limit for the parameter β_0. We can see that the estimated value of β_0 becomes negative when the reference input is 40 °C/sec. That leads to a saturation of the control.

In Fig. 6.4.7 zero is given as the lower limit. Therefore the bad behavior shown in Fig. 6.4.6 is not seen. The steady-state error is reduced by the integrator, but it still remains because the estimated parameter β_0 reaches the lower limit, zero.

As seen above, the integrator plays an important role to eliminate the steady-state error. We also see that it is crucial to give a reasonable boundary to the parameter β_0. However, it is hard to determine precisely the lower value of β_0 in the case of type 1 structure. The control using type 1 model structure may accommodate the variation of the time-delay. However, if the actual time-delay is much larger than that used in the model, we simply say β_0 is essentially equal to zero. Therefore we cannot determine the lower limit other than zero.

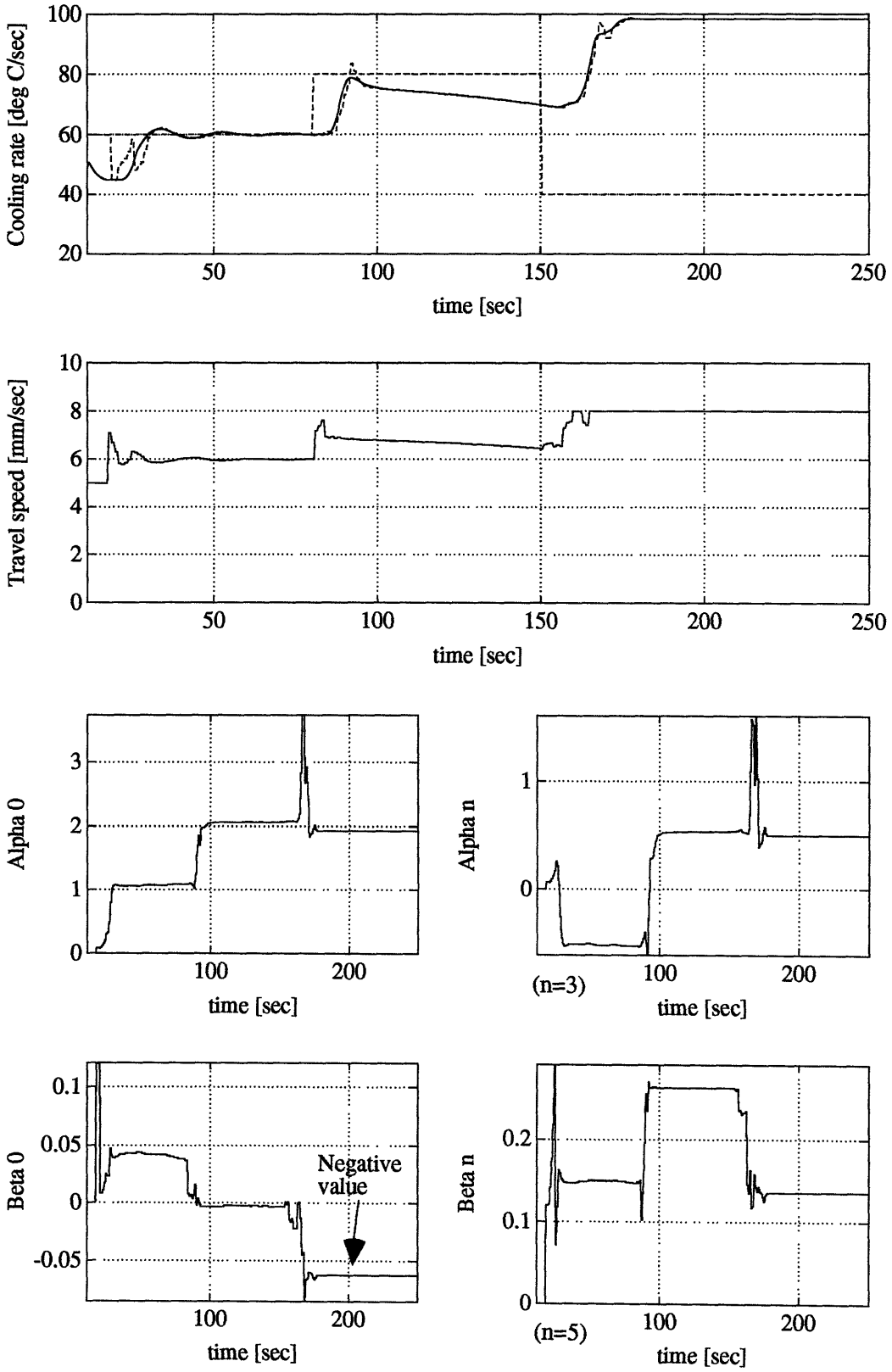


Fig. 6.4.6 Weighted one-step-ahead control with integrator (No limit on β_0)

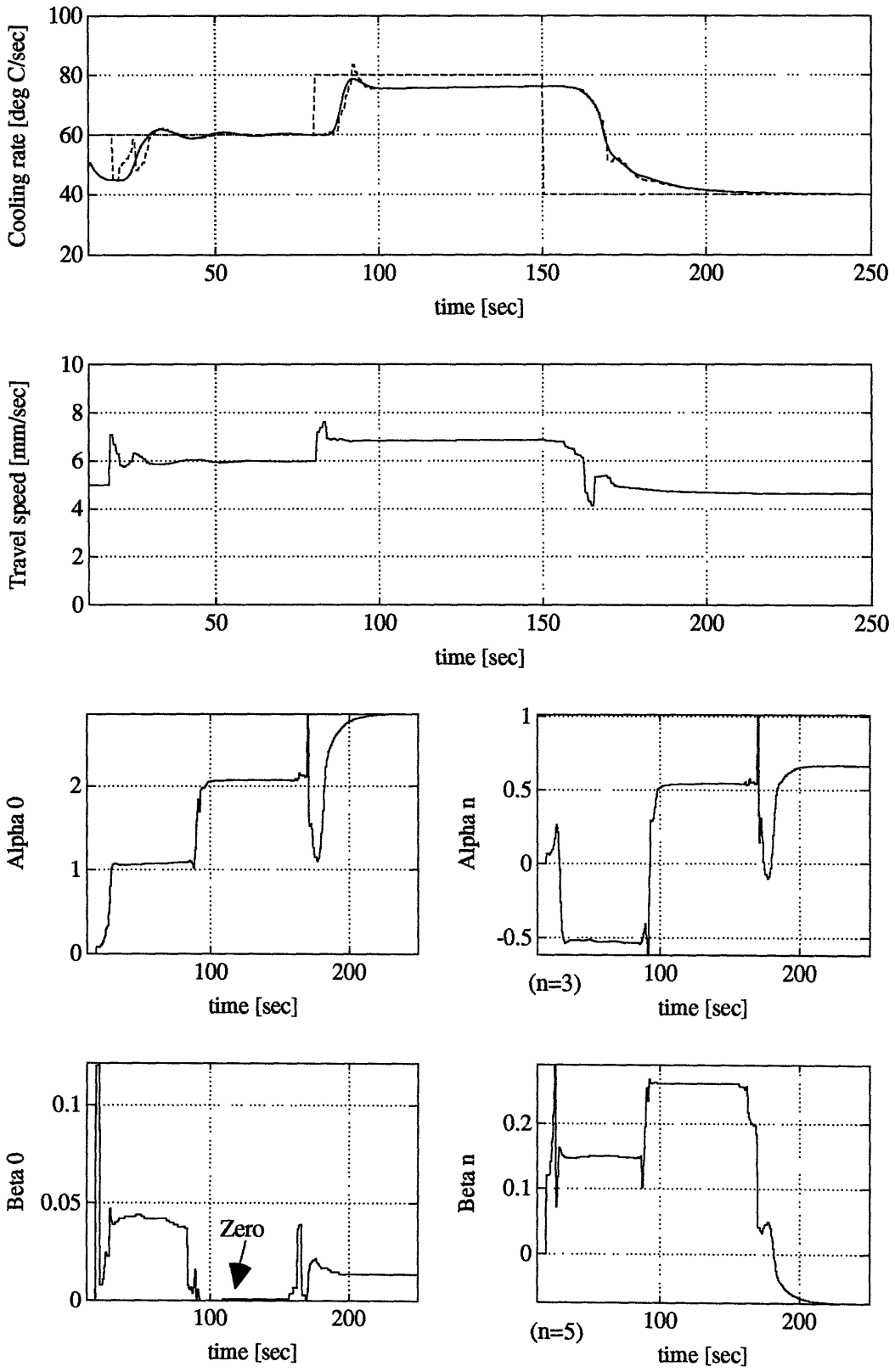


Fig. 6.4.7 Weighted one-step-ahead control with integrator ($\beta_0 \geq 0$)

6.4.2 Performance of Type 2 model structure

Figures 6.4.8 and 6.4.9 show the response by the weighted-one-step-ahead adaptive control which has type 2 model structure. In both cases we take 0.03 as the weight λ. As the factor of the integrator we use 0.7 in Fig. 6.4.8 and 0.8 in Fig. 6.4.9. The value of 0.01 is given as the lower limit of the estimate of β_0. The lower limit is determined by a simple calculation shown below.

In both figures the cooling rate reaches the given input reference, and there is no steady-state error. An overshoot takes place when the reference input is 80 °C/sec. However, it converges to the reference quite fast. The settling time is about 20 sec, which is about half of that in the PI control case. More comparison for various inputs will be made later.

The parameter estimates frequently change their values, because the model structure used for the control switches from one to another as shown in Fig. 6.4.10. However, the input given to the plant does not change so much as the parameters vary.

The over all performance of this control seems to be better than PI control. The controller produces a step-wise input signal as soon as the reference input changes. That results in a faster response.

(1) Lower limit of the parameter β_0

In type 2, we assume the delay-free part of the process is a second order system. The second order system is used simply to accommodate a gentler rising just after the step input is given. Therefore, the second order system would have the form:

$$\frac{abK}{(s+a)(s+b)} \qquad \text{where } a \text{ and } b \text{ is positive real.} \qquad (6.4.1)$$

We also assume $a << b$. Therefore the mode associated with the parameter a is slower than that associated with the parameter b.

The z transformation of the transfer function yields

$$Z(\frac{1-\exp(-sT)}{s}\frac{abK}{(s+a)(s+b)}) = \frac{Az^{-1}+Bz^{-2}}{(1-\exp(-aT)z^{-1})(1-\exp(-bT)z^{-1})} \quad (6.4.2)$$

where

$$A = k\frac{(b-a)-b\exp(-aT)+a\exp(-bT)}{(b-a)}$$

$$B = k\frac{a\exp(-aT)(1-\exp(-bT))-be(-bT)(1-\exp(-aT))}{(b-a)}$$

We are concerned with the value of A. Since $a << b$ we could roughly approximate the value of A as:

$$A = k(1-\exp(-aT)) \quad (6.4.3)$$

The observation of the step response shows the longest time constant of the system is about 10 sec and the gain between the scaled input and the scaled output is 0.2. Therefore we obtain the values:

$$a = 1/10$$
$$k = 0.2 \quad (6.4.4)$$

Using the above values, the equation shows $A = 0.019$.
The value of β_0 is equal to the value of A. Therefore 0.01 can be taken as the lower limit of β_0.

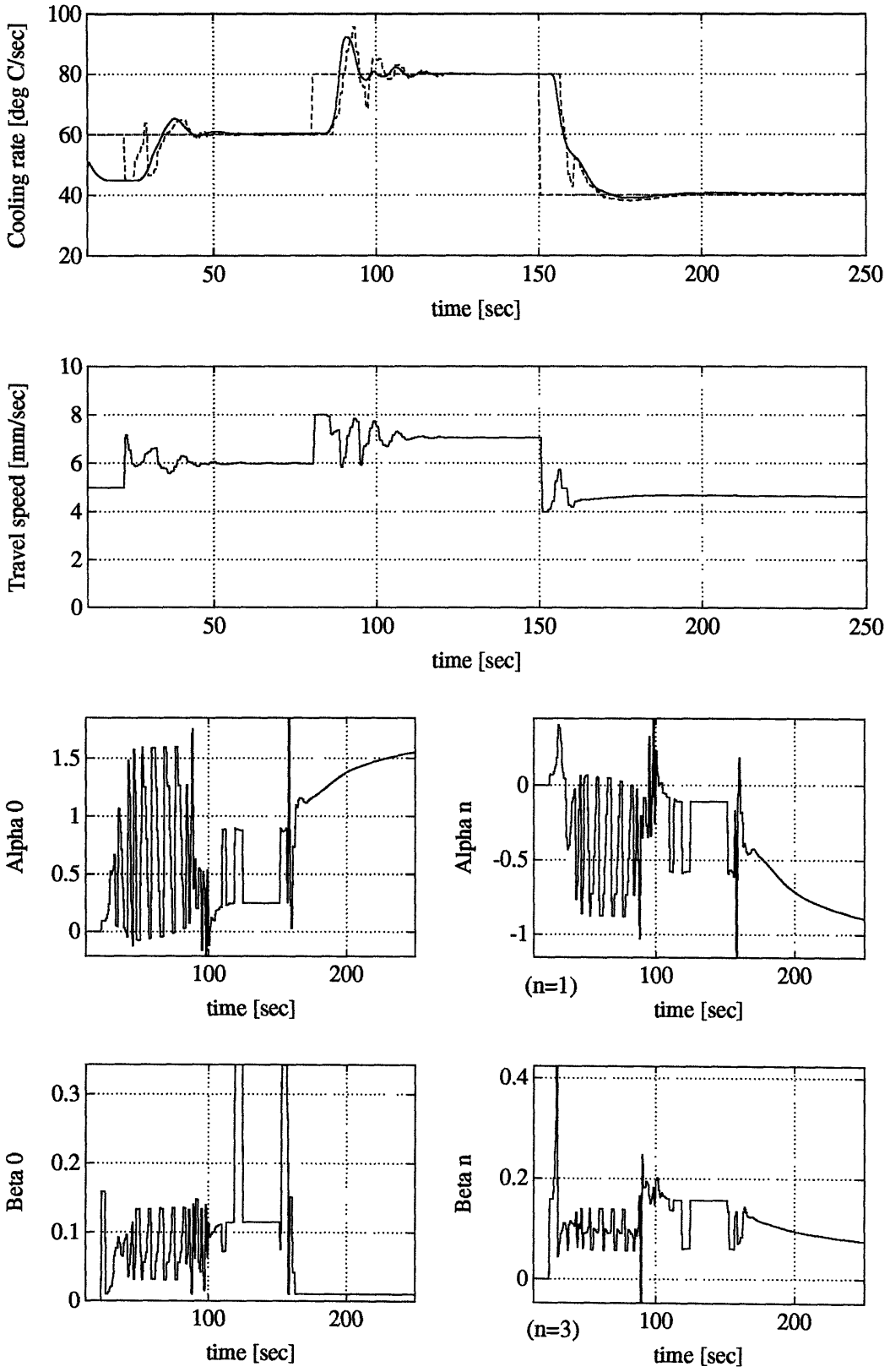


Fig. 6.4.8 Weighted one-step-ahead control with type 2 structure ($\lambda = 0.03,\ \gamma = 0.7$)

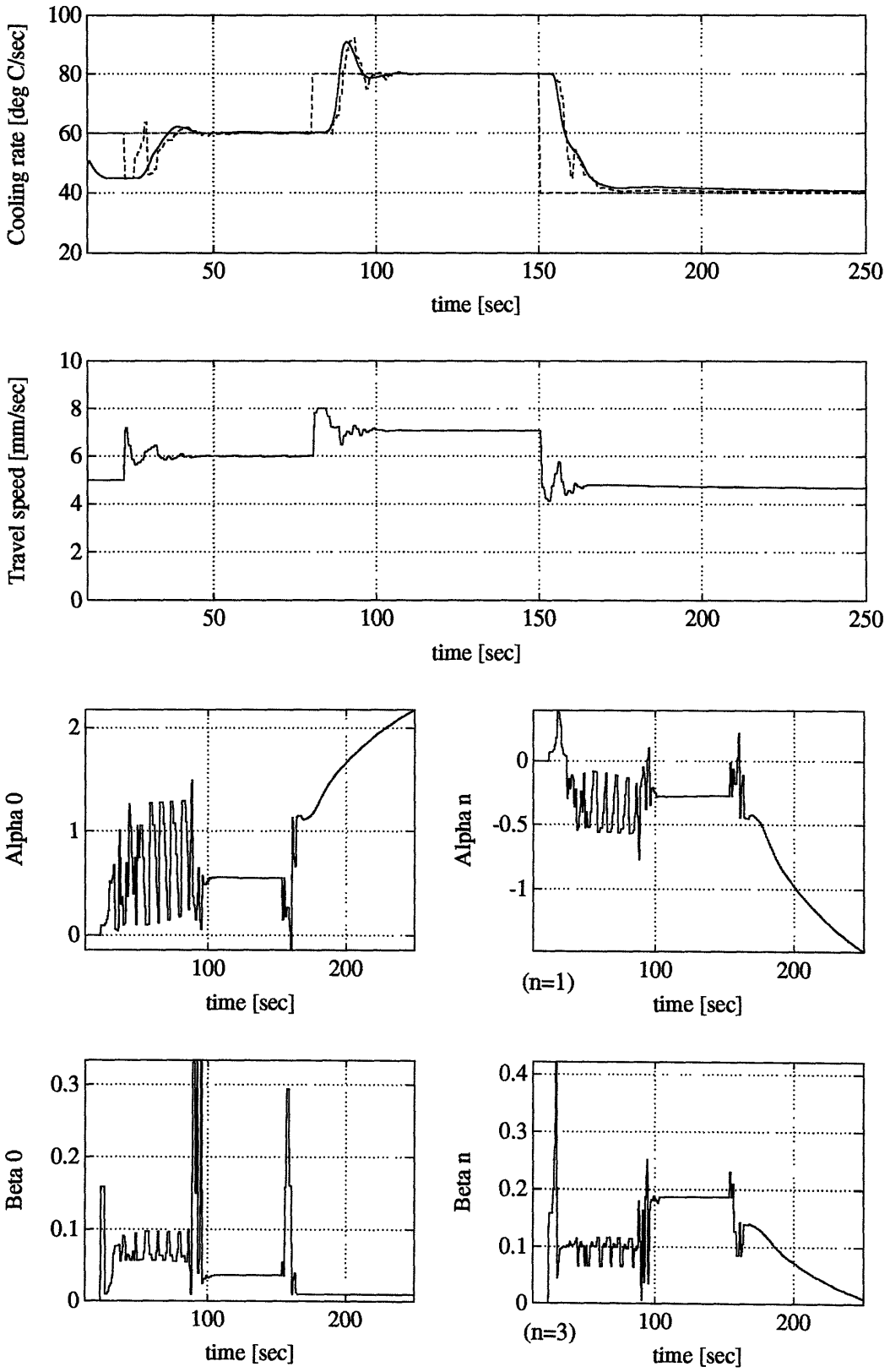


Fig. 6.4.9 Weighted one-step-ahead control with type 2 structure ($\lambda = 0.03,\ \gamma = 0.8$)

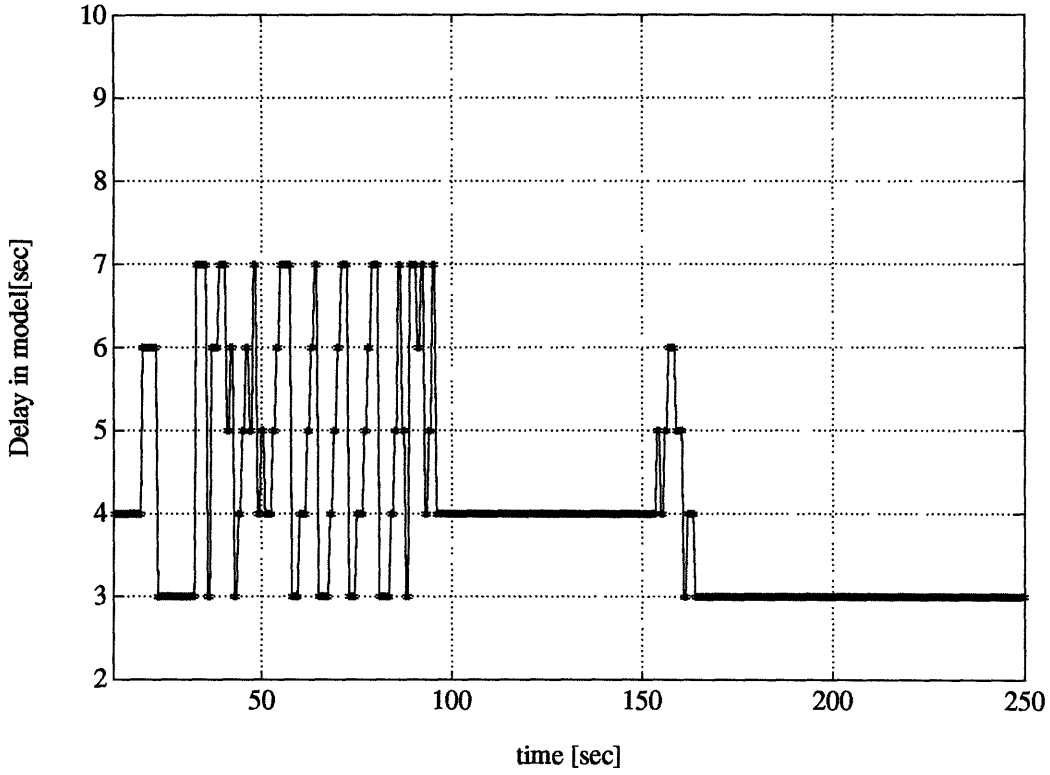


Fig. 6.4.10 Change of model structure ($\lambda = 0.03$, $\gamma = 0.8$)

(2) Comparison between weighted-one-step-ahead control and PI control

Figure 6.4.11 shows the comparison between the responses of the weighted-one-step-ahead control with type 2 structure and PI control. The same values as used in Fig. 6.4.9 are taken for the factors associated with the control. As mentioned above, the weighted-one-step controller achieves a faster response than PI controller. The settling time is about half of that of the PI controller.

Although the stability is not guaranteed from a theoretical point of view, any unstable behavior does not take place. The model structure used for the control switches over from one to another as shown in Fig. 6.4.12. However, no bad behavior is caused by such frequent switch-over of the models.

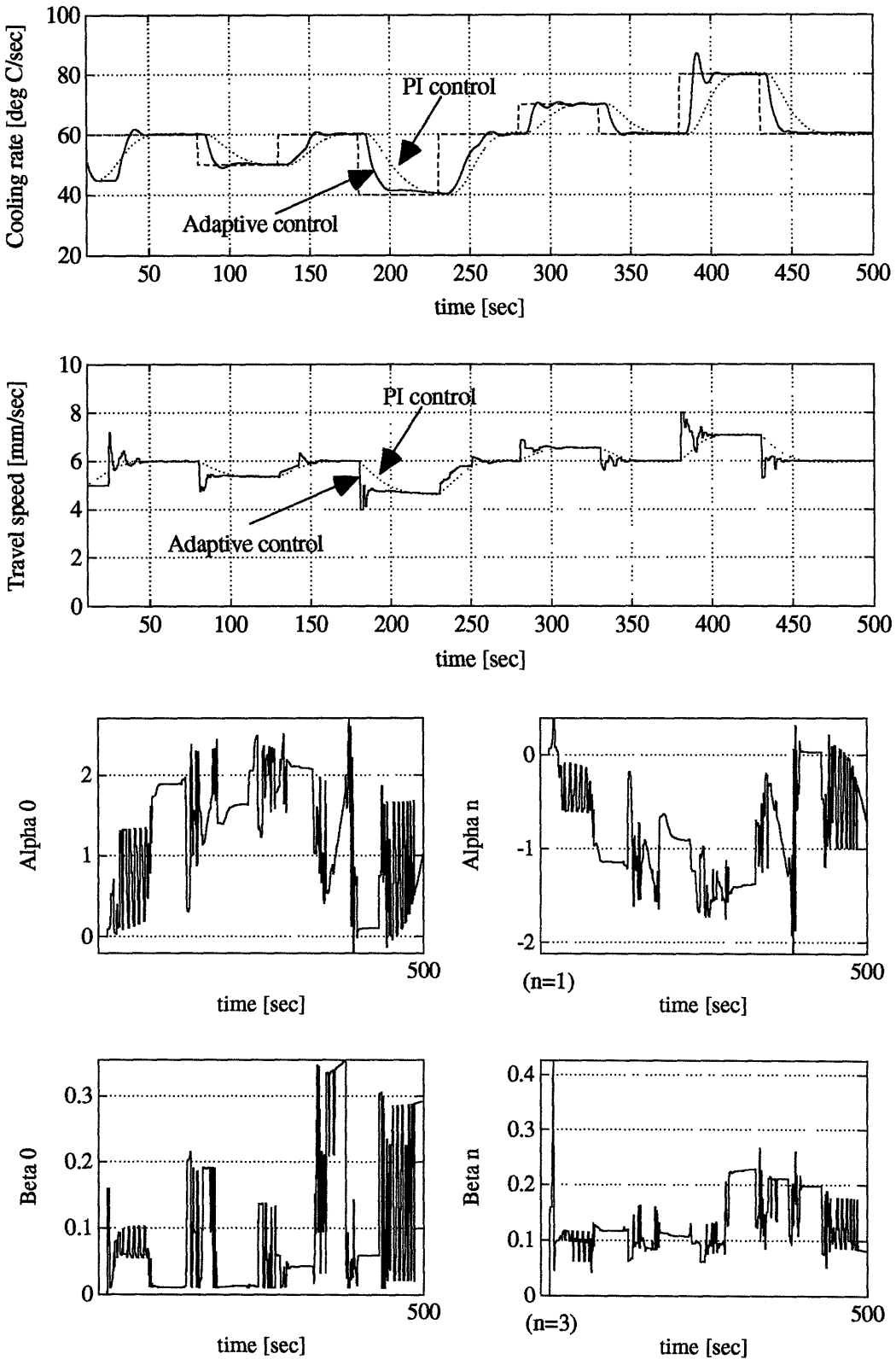


Fig. 6.4.11 Comparison between one-step-ahead-control and PI control
($\lambda = 0.03,\ \gamma = 0.8$)

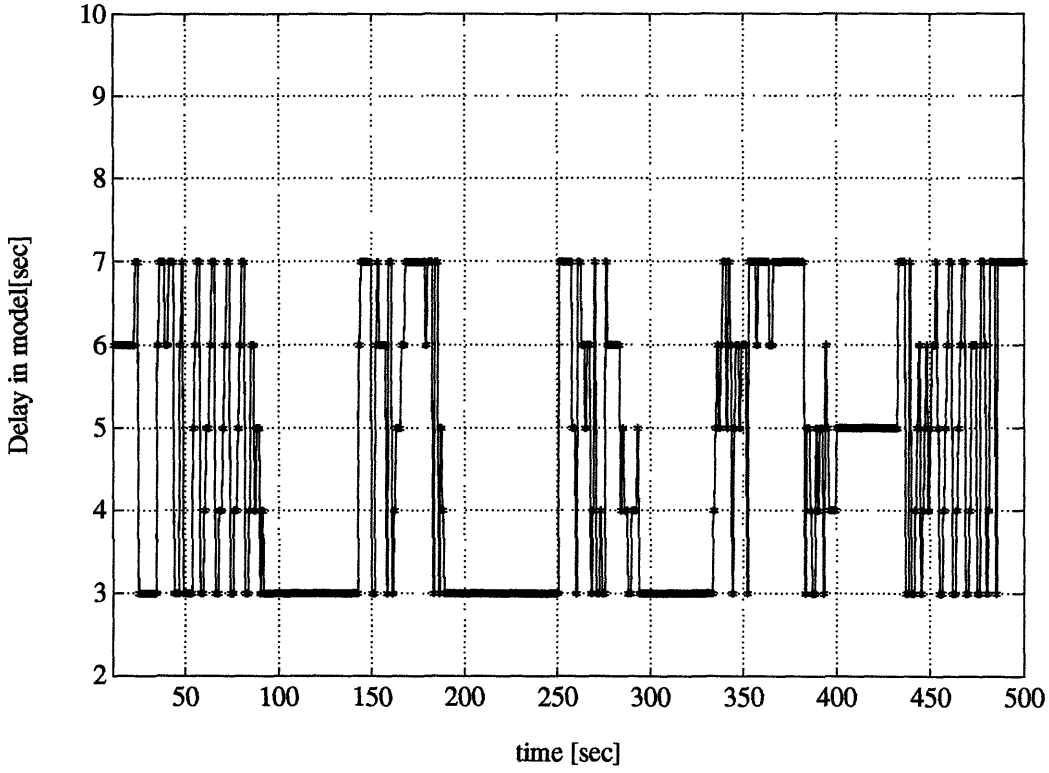


Fig. 6.4.12 Change of model structure

In Fig. 6.4.11, the oscillatory control action is observed when the reference input is changed. The frequent oscillation is sometimes unfavorable for the machine. To avoid this, the weight λ needs to be increased. Figure 6.4.13 shows the response when λ and γ take 0.1 and 0.5 respectively. The increase of λ increases the stead-state error, and requires more integral action. Therefore, γ has smaller value than the case in Fig. 6.4.11. The figure shows that the oscillation is reduced, but the response is slower that in Fig. 6.4.11. There is the case where the response is even slower than that of PI control. We can also see some rather big overshoots. Thus, when we use the one-step-ahead control scheme, there is a trade-off between the performance and the control action.

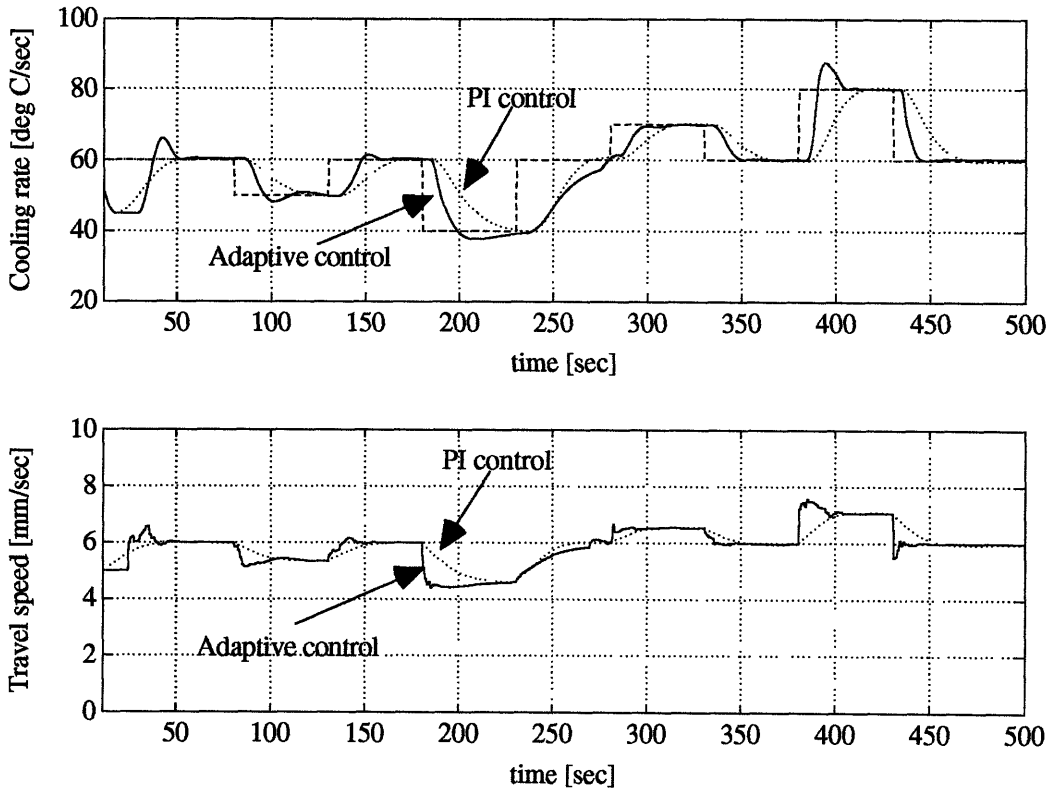


Fig. 6.4.13 One-step-ahead-control and PI control ($\lambda = 0.1$, $\gamma = 0.5$)

6.5 Concluding remarks

The objectives of using adaptive control are to deal with the nonlinearity of the system. Therefore, the rapid adaptation to the change of the system structure and the parameters is crucial. From the point of view, the recursive least square algorithm is quite satisfactory.

Two types of model structures for the weighted-one-step-ahead adaptive control are proposed to cope with a varying time-delay. Since the type 1 structure completely violates the conditions for successful adaptive control, it fails to control the cooling rate. On the other hand, the type 2 structure succeeds in it and shows a quite good performance.

The simulation shows the control with the type 2 model structure performs better than PI controller. The settling time can be reduced to almost a half of that of PI control. From the theoretical point of view, the stability is not guaranteed. However, we do not observe any unstable behavior in the simulation.

It depends on the nature of the plant whether one step-ahead adaptive control works. For the cooling rate control its applicability is supported by the fact:

1. The system can be approximated by a first order system or a second order system with a relative degree 2.

2. Unmodeled dynamics is negligible. The system does not have a resonance in a higher frequency.

Feature **2** is crucial when we use this control scheme. If not, we would have to degrade the performance of this control due to a possible excitement of an undesirable high frequency mode, which may produce an unfavorable behavior. Thus, the merit of the adaptive control would diminish.

The constraints given to the parameters play a key role for this control scheme. From the theoretical point of view the convergence of the output to a desired value is guaranteed if the system is time-invariant and linear. However, the convergence of the parameter estimates to the actual values is not necessarily guaranteed. (See the discussions by Slotine et al. (1991).) In our problem the system is nonlinear and more complicated. Therefore there is a big possibility that the parameter estimates are driven to undesirable directions, even if the output prediction by these parameters meets with the sequence of the actual output data. If we use the parameters without any consideration, the control may result in an unfavorable result. In particular the parameter β_0 is quite important since it determines whether the input should be increased or decreased when there is an error between the output and the reference input. The only way to avoid a failure of the control is to place a constraint on the parameter estimation.

It is obvious from the nature of the adaptive control that it can cope with the problem for the unknown plant perturbation, which is a typical problem in manufacturing systems. In the simulation only the performance for the step response is shown, although we are particularly concerned with a disturbance rejection. However, the performance for the disturbance rejection is closely related to the performance for the command following. Therefore better step responses result in better rejection of the disturbance in a low frequency. From the nature of heat transfer we do not have to consider a disturbance in a higher frequency. Therefore the control must show a good performance for the disturbance rejection. Further discussion will be found in Chapter 8.

One step-ahead adaptive control sometimes brings about the oscillatory control action, which may be unfavorable from the view point of maintaining the machine. To avoid the oscillation, we have to sacrifice the performance of the control. Therefore, there is a trade-off between the performance and the control action.

Chapter 7

Application of Control Using Neural Networks to Cooling Rate

In this chapter we discuss the application of control using neural networks to the cooling rate. The issues of our discussion are as follows.

1. How accurately can neural networks approximate nonlinear systems?
2. How fast can neural networks identify nonlinear dynamic systems?
3. What is the drawbacks of neural networks?
4. What kind of control scheme can be used for the cooling rate?
5. How can we design the control?
6. How does the control perform?

Neural networks are quite attractive from the viewpoint that we can approximate any nonlinear system. However they will have limitations like any other methods. Therefore first we have to know the limitations. Then we design the control, considering the limitations.

Section 7.1 reviews an error back-propagation method, which is widely used for the identification of the parameters in multi-layer neural networks. We also present back-propagation with a distal system, which is used for the design of the control using neural networks. Section 7.2 deals with nonlinear identification problems. In particular we pay attention to two identification methods, series-parallel identification and parallel identification. The parallel identification based on a conjugate gradient method is proposed. The performance of the approximation and the identification by neural networks are discussed in Section 7.3. Based on the discussion in Section 7.3, a feasible control structure using neural networks is discussed in Section 7.4. In particular we pay attention to Smith's principle, taking into consideration the limitations of neural networks. Section 7.5 deals with the identification of the nonlinear model for the cooling rate. In Section 7.6 we discuss the control using neural networks for the cooling rate. The performance of the control is also discussed there. The concluding remarks are made in Section 7.7.

7.1 Back-Propagation method

7.1.1 Algorithm of back-propagation

The recent advancement in the computation by neural networks is due to the algorithm of an error back-propagation method. The basis of the back-propagation is simply an incremental gradient descent method discussed in Section 5.4.2. As mentioned in Section 5.3, multilayer neural networks have a structure:

$$o_i^{[n]} = \varphi(net_i^{[n]}) \qquad i = 1 \cdots num[n] \tag{7.1.1a}$$

$$net_i^{[n]} = \sum_{j=1}^{num[n-1]} w_{ij}^{[n]} o_j^{[n-1]} \qquad i = 1 \cdots num[n] \tag{7.1.1b}$$

$$O_i = o_i^{[N]} \qquad i = 1 \cdots num[N] \tag{7.1.1c}$$

$$o_i^{[0]} = I_i \qquad i = 1 \cdots num[0] \tag{7.1.1d}$$

$$n = 1 \cdots N$$

where $o_i^{[n]}$ denotes the output of the i-th cell in the n-th layer, $\varphi()$ denotes a nonlinear function the cell, $w_{ij}^{[n]}$ is the weight from the j-th cell in the (n-1)-th layer to the i-th cell in the n-th layer, O_i is the i-th output of the neural network, and I_i is the i-th input of the neural network, and $num[n]$ denotes the number of the cells in the n-th layer.

The typical cost function is given by:

$$E(\boldsymbol{w}) = \sum_{l=1}^{num[N]} \frac{1}{2}(T_l - O_l(\boldsymbol{I}))^2 \qquad \boldsymbol{w} = \{w_{ij}^{[n]}\}_{i,j,n} \tag{7.1.2}$$

where $\{\boldsymbol{T},\boldsymbol{I}\}_t = \{T_1 \cdots T_{num[N]}, I, \cdots, I_{num[0]}\}_t$ is a t-th sample data in which T_l denotes a sample data for the output and I_l is a sample data for the input. The rule updating the parameters is given by the gradient descent method:

$$\Delta w_{ij}^{[n]} = -\eta \frac{\partial E(\boldsymbol{w})}{\partial w_{ij}^{[n]}} \qquad \eta\text{: Step size} \tag{7.1.3}$$

Therefore, our task is to calculate the derivative $\partial E(\boldsymbol{w})/\partial w_{ij}^{[n]}$. Using the chain rule and assuming $\varphi()$ is differentiable, we can get

$$\frac{\partial E(w)}{\partial net_i^{[n]}} = \sum_k \frac{\partial E(w)}{\partial net_k^{[n+1]}} \frac{\partial net_k^{[n+1]}}{\partial o_i^{[n]}} \varphi'(net_i^{[n]}) = \sum_k \frac{\partial E(w)}{\partial net_k^{[n+1]}} w_{ki}^{[n+1]} \varphi'(net_i^{[n]}) \tag{7.1.4}$$

The introduction of a new parameter $\delta_i^{[n]} (= \partial E(w)/\partial net_i^{[n]})$ rewrites (7.1.4) as

$$\delta_i^{[n]} = \sum_k \delta_k^{[n+1]} w_{ki}^{[n+1]} \varphi'(net_i^{[n]}) \tag{7.1.5}$$

Since $\delta_i^{[N]}$ in the final layer is given by

$$\delta_i^{[N]} = \frac{\partial E(w)}{\partial net_i^{[N]}} = \frac{\partial}{\partial net_i^{[N]}} \left(\sum_{l=1}^{num[N]} \frac{1}{2} (T_l - O_l)^2 \right) = (T_i - O_i) \varphi'(net_i^{[N]}) \tag{7.1.6}$$

we can calculate $\delta_i^{[n]}$ backward from the final layer to the first layer by (7.1.5) since we know the function $\varphi()$. Finally we can obtain $\partial E(w)/\partial w_{ij}^{[n]}$ by

$$\frac{\partial E(w)}{\partial w_{ij}^{[n]}} = \frac{\partial E(w)}{\partial net_i^{[n]}} \frac{\partial net_i^{[n]}}{\partial w_{ij}^{[n-1]}} = \delta_i^{[n]} o_j^{[n-1]} \tag{7.1.7}$$

Obviously we have to calculate $\varphi'(net_i^{[N]})$ to get $\delta_i^{[N]}$ by (7.1.4). To do so, first we calculate them forward from the first layer to the final layer by (7.1.1), and then we calculate $\delta_i^{[n]}$ backward. The equations (7.1.5) and (7.1.6) show the error looks like to be back-propagated from the final layer to the first layer. That is the reason the algorithm is called "Error back-propagation".

As the function $\varphi()$, the logistic function $1/(1+\exp(-x))$, the sigmoid function $(1-\exp(-x))/(1+\exp(-x))$ and $\tanh(x)$ are widely used. Thus, the calculation of $\varphi'(net_i^{[N]})$ is quite easy.

7.1.2 Back-propagation with a distal system

In this section we deal with the minimization problem where we have a distal system as shown in Fig. 7.1.1. The objective is to find the neural network such that it minimizes the cost function:

$$E(\boldsymbol{w}) = \sum_{l=1}^{L} \frac{1}{2}(T_l - Y_l)^2 \tag{7.1.8}$$

where Y_l is the output of the distal system, not the output of the neural network. We can see the neural network becomes an inverse model of the distal system.

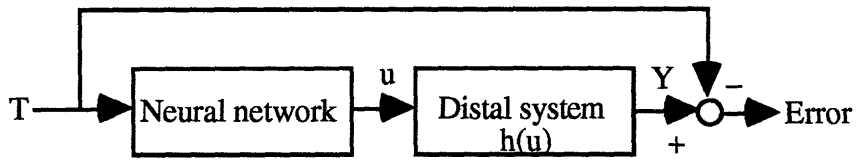


Fig. 7.1.1 Neural network with a distal system

For this minimization problem, we can also apply a back propagation method if the distal system is differentiable and we also can manipulate explicitly the differentiation of the distal system.

Instead of (7.1.6) in the back-propagation algorithm, we use the equation:

$$\begin{aligned}\delta_i^{[N]} &= \frac{\partial E(\boldsymbol{w})}{\partial net_i^{[N]}} = \frac{\partial}{\partial net_i^{[N]}}\left(\sum_{l=1}^{L}\frac{1}{2}(T_l - Y_l)^2\right) = \sum_{l=1}^{L}(T_l - Y_l)\sum_{i=1}^{num[N]}\frac{Y_l}{\partial u_i}\frac{\partial u_i}{\partial net_i^{[N]}} \\ &= \sum_{l=1}^{L}(T_l - Y_l)\sum_{i=1}^{num[N]}\frac{h_l(\boldsymbol{u})}{\partial u_i}\varphi'(net_i^{[N]})\end{aligned} \tag{7.1.9}$$

where $\boldsymbol{h}(\boldsymbol{u}) = (h_1(\boldsymbol{u})\cdots h_L(\boldsymbol{u}))$ is the distal system with the input $\boldsymbol{u} = (u_1 \cdots u_{num[N]})$

It is sometimes the case that we cannot explicitly calculate the derivative of the distal system. In this case, we can resort to neural networks again. The idea is first to make the model for the distal system by a neural network, and then to create the second neural network by the method (7.1.9). Therefore, the function $\boldsymbol{h}(\boldsymbol{u})$ in (7.1.8) is also a neural network. In this formation the derivative of the neural network describing the distal system can be calculated in the similar manner to the back-propagation by the chain rule.

The design of the nonlinear controller using neural networks is based on the minimization of the cost function represented by the performance of the resulting controlled system. The controller optimizes the performance via the plant dynamics. Therefore, the framework of this problem is essentially parallel to the back-propagation

problem with a distal system. The back-propagation with a distal system will be used for the design of the nonlinear controller for the cooling rate in Section 7.6.1.

7.2 Identification by neural networks

7.2.1 Series-parallel identification and Parallel identification

As discussed in Section 5.3.2, Narendra et al. (1990) proposed two types of identification methods, parallel identification and series-parallel identification. They are given by the identification structures:

Parallel identification

$$\hat{y}(t+1) = NN[\hat{y}(t), \hat{y}(t-1), \cdots, \hat{y}(t-n+1), u(t), u(t-1), \cdots, u(t-m+1)] \qquad (7.2.1)$$

Series-parallel identification

$$\hat{y}(t+1) = NN[y(t), y(t-1), \cdots, y(t-n+1), u(t), u(t-1), \cdots, u(t-m+1)] \qquad (7.2.2)$$

where $y(t)$ is the output of the plant, and $\hat{y}(t)$ is the output of the model.
We identify the parameters so that the cost function

$$E = \sum_t (y(t) - \hat{y}(t))^2 \qquad (7.2.3)$$

is minimized.
The model structures are same for both cases, but how the parameters are identified is different . The results by both methods seem to be identical. However, there are the following big differences.

1. We cannot use back-propagation for the parallel identification. Suppose we change the parameters by back-propagation to reduce the output error. We easily see the change of the parameters also brings about the change of all the values of $\hat{y}(k), \hat{y}(k-1), \cdots, \hat{y}(k-n+1)$ because they are also the estimates by the neural network. Therefore, back-propagation does not necessarily reduce the output error.
2. By similar reasoning we can see that back-propagation can be used for the series-parallel identification.

7.2.2 Series-parallel estimation and parallel estimation

Two types of identification problems in the previous section also suggests the following forms of estimation:

Parallel estimation
The parallel estimate $\hat{y}(t)$ is calculated by:

$$\hat{y}(t+1) = NN[\hat{y}(t), \hat{y}(t-1), \cdots, \hat{y}(t-n+1), u(t), u(t-1), \cdots, u(t-m+1)] \qquad (7.2.1)$$

given the initial state $[\hat{y}(n-1), \cdots, \hat{y}(0)]$

Series-parallel estimation
The series-parallel estimate $\hat{y}(t)$ is calculated by:

$$\hat{y}(t+1) = NN[y(t), y(t-1), \cdots, y(t-n+1), u(t), u(t-1), \cdots, u(t-m+1)] \qquad (7.2.2)$$

The parallel identification creates a neural network that produces a good parallel estimate. The series-parallel identification yields a model with good performance for the series-parallel estimate. However, we can not say anything about the performance of the parallel estimate when we take series-parallel identification, because an estimate error at a time would yield the error of the parallel estimate successively from the time on.

To see the above statement more clearly, consider the following simpler linear plant:

$$y(t+1) = 0.5y(t) + 0.5u(t) \qquad 0 \le u(t) \le 1$$

Then $0 \le y(t) \le 1$. Now let us suppose that we want to make a model of the above plant, which has the square error ($(y(t) - \hat{y}(t))^2$) less than 0.0001, and we get the model that is given by:

$$y(t+1) = 0.49y(t) + 0.5u(t)$$

When we use this model as the series-parallel estimation model:

$$\hat{y}(t+1) = 0.49y(t) + 0.5u(t)$$

the square error is always less than 0.0001 because the output of the plant is within the range from 0 to 1.0 and the parameter error is 0.01. Therefore, the model is sufficient for the series-parallel estimation. However, if we use the model as the parallel estimation:

$$\hat{y}(t+1) = 0.49\hat{y}(t) + 0.5u(t)$$

the square error is not always less than 0.0001. For example, consider the steady-state response when $u(t) = 1$. Then the plant output $y(t)$ is equal to 1.0, but the estimate $\hat{y}(t)$ is equal to 0.98 $(= 0.5 / 0.51)$. Therefore the error is 0.0004, and the model does not satisfy the given condition. We need a more accurate model for the parallel estimation. If the system is nonlinear, the nonlinear model would bring about a bigger error for the parallel estimation, while it would suffice for the series-parallel estimation.

Next we pay attention to the identification. As discussed in Section 5.4.2, the nonlinear model is identified by minimizing the quadratic error. The small error yields the small adjustment of the model parameters. Suppose we use the series-parallel identification, and eventually we get the model that yields a favorably small quadratic error, which would almost halt the adjustment of the parameters. However it may not suffice the parallel estimation for the above-mentioned reason. In other words the series-parallel identification tends to be more insensitive to the adjustment of the parameters than the parallel identification. Therefore, if we need a model sufficient for the parallel estimation, it would be better to use the parallel identification.

The difference between the series-parallel identification and the parallel identification is quite crucial to the design of a controller. If we use one step-ahead predictive control and an adaptive control scheme, we may use a series-parallel identification, because the control only requires an accurate prediction at the one step ahead by using the present and past output of the plant.

However, if we use Smith's method, the control requires an accurate estimate at all times without referring to the actual output of the plant. It implies that the model must show good performance for the parallel estimation, and the series-parallel identification may not work for this problem.

7.2.3 Conjugate gradient method for parallel identification

As mentioned above, we cannot apply an incremental identification method for the parallel identification problem. Therefore we will resort to a batch identification method. There is no reason we have to use the descent gradient method for the batch identification, because it is the most inefficient for the minimization problem.

In Section 5.4.2, we discussed some methods for the nonlinear minimization problem. Above them the conjugate gradient is thought of as the most efficient method. (See the discussion by Press et al. (1992)) Therefore, we will use the conjugate gradient method for the parallel identification.

We have already discussed the algorithm of the conjugate gradient method. Therefore, we will constrain our interest on how to get the derivative of the cost function because the conjugate gradient method requires it. The manipulation is rather complicated, as shown below.

Consider SISO system which is given by

$$\begin{aligned} &\hat{y}(t+1) = NN(\hat{y}(t),\cdots,\hat{y}(t-l),u(t),\cdots,u(t-m),\boldsymbol{w}) \\ &\boldsymbol{w} = \{w_{ij}^{[n]}\}_{1\le i\le num[n],1\le j\le num[n-1],1\le n\le N} \end{aligned} \tag{7.2.4}$$

The neural network takes the form (7.1.1), where $num[N]=1$, and $num[0]=l+m+2$. The cost function is given by

$$\begin{aligned} &E(\boldsymbol{w}) = \sum_{t=l+1}^{T}(y(t)-\hat{y}(t))^2 \\ &\text{given the initial state } \hat{y}(0)=y(0),\cdots,\hat{y}(l)=y(l) \end{aligned} \tag{7.2.5}$$

Therefore the derivative of the cost function can be written as:

$$\frac{\partial E(\boldsymbol{w})}{\partial w_{ij}^{[n]}} = \sum_{t=l+1}^{T}(y(t)-\hat{y}(t))\frac{\partial \hat{y}(t)}{\partial w_{ij}^{[n]}} \tag{7.2.6}$$

The notation $\partial\hat{y}(t)/\partial w_{ij}^{[n]}$ is rather confusing, because it has two interpretations:

(1) The ordinal derivative

(2) The derivative including the whole effect of $\partial\hat{y}(l+1)/\partial w_{ij}^{[n]}$

To avoid the confusion, we will use the notation:

$$\frac{\partial^+ \hat{y}(t)}{\partial w_{ij}^{[n]}} \tag{7.2.7}$$

for the latter case.

Therefore we must rewrite the equation (7.2.6) as

$$\frac{\partial E(\boldsymbol{w})}{\partial w_{ij}^{[n]}} = \sum_{t=l+1}^{T} (y(t) - \hat{y}(t)) \frac{\partial^+ \hat{y}(t)}{\partial w_{ij}^{[n]}} \tag{7.2.8}$$

Considering the model structure (7.2.4), we can easily see $\partial^+ \hat{y}(t) / \partial w_{ij}^{[n]}$ is given by

$$\frac{\partial^+ \hat{y}(t)}{\partial w_{ij}^{[n]}} = \frac{\partial \hat{y}(t)}{\partial w_{ij}^{[n]}} + \sum_{\lambda=1}^{l+1} \frac{\partial \hat{y}(t)}{\partial \hat{y}(t-\lambda)} \frac{\partial^+ \hat{y}(t-\lambda)}{\partial w_{ij}^{[n]}} \tag{7.2.9}$$

Using the chain rule, we can obtain

$$\begin{aligned}
&\frac{\partial \hat{y}(t)}{\partial w_{ij}^{[n]}} = \frac{\partial \hat{y}(t)}{\partial net_i^{[n]}} o_j^{[n-1]} = \xi_i^{[n]} o_j^{[n-1]} \\
&\frac{\partial \hat{y}(t)}{\partial \hat{y}(t-\lambda)} = \frac{\partial \hat{y}(t)}{\partial o_\lambda^{[0]}} = \sum_k \frac{\partial \hat{y}(t)}{\partial net_k^{[1]}} w_{k\lambda}^{[1]} = \sum_k \xi_k^{[1]} w_{k\lambda}^{[1]} \\
&\xi_i^{[n]} = \frac{\partial \hat{y}(t)}{\partial net_i^{[n]}(t)} = \sum_k \frac{\partial \hat{y}(t)}{\partial net_k^{[n+1]}(t)} w_{ki}^{[n+1]} \varphi'(net_i^{[n]}(t)) = \sum_k \xi_k^{[n+1]} w_{ki}^{[n+1]} \varphi'(net_i^{[n]}(t))
\end{aligned} \tag{7.2.10}$$

where $o_\lambda^{[0]} = \hat{y}(t-\lambda) \quad \lambda = 1, \cdots, l+1$

Therefore, we can determine $\partial \hat{y}(t) / \partial w_{ij}^{[n]}$ and $\partial \hat{y}(t) / \partial \hat{y}(t-\lambda)$ in (7.2.9) via the variables $\xi_i^{[n]}$ by the algorithm (7.2.10) which is similar to the back-propagation. Thus we can get $\partial^+ \hat{y}(t) / \partial w_{ij}^{[n]}$ by iteratively summing $\partial \hat{y}(t) / \partial w_{ij}^{[n]}$ and $\partial \hat{y}(t) / \partial \hat{y}(t-\lambda)$ for all the sample data as shown in (7.2.9). Finally $\partial E(\boldsymbol{w}) / \partial w_{ij}^{[n]}$ can be obtained by (7.2.8).

The computation above is executed every time the conjugate gradient algorithm searches the line minimum. Therefore it needs a large number of computations. However there is no better method than this for the parallel identification, and we will see later that this method is superior to the back-propagation.

7.3 Approximation and identification by neural networks

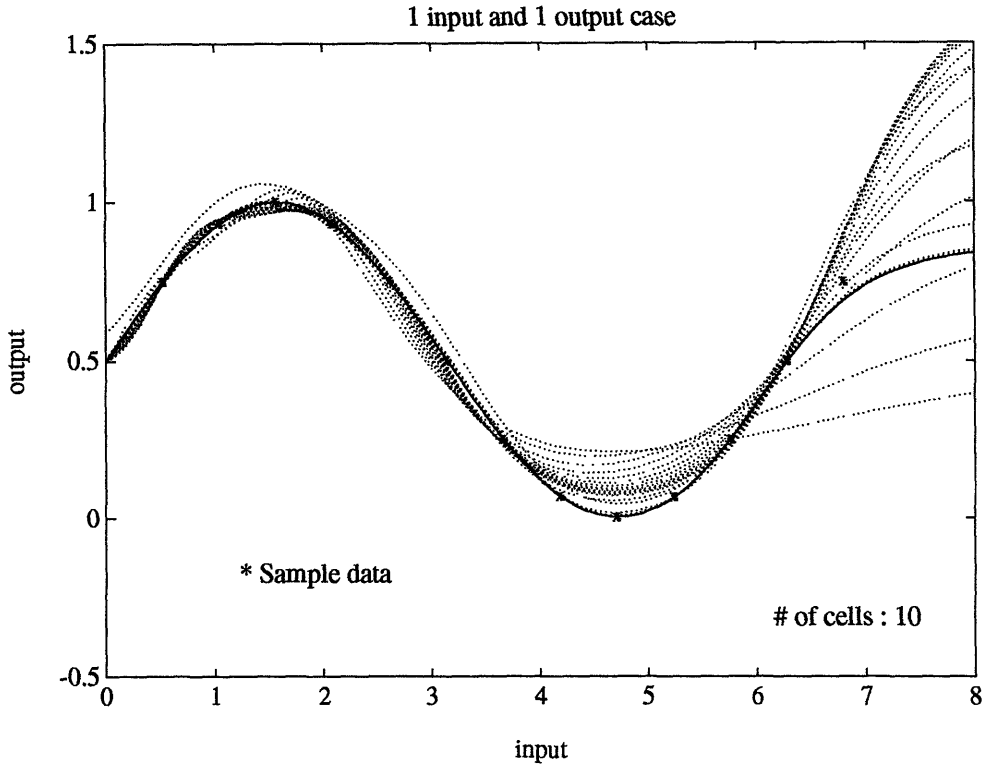
In this section we discuss how neural networks behave for nonlinear problems. Section 7.3.1 deals with the approximation of nonlinear systems by neural networks. In Section 7.3.2 we discuss the identification of nonlinear dynamics by neural networks.

7.3.1 Approximation of nonlinear systems

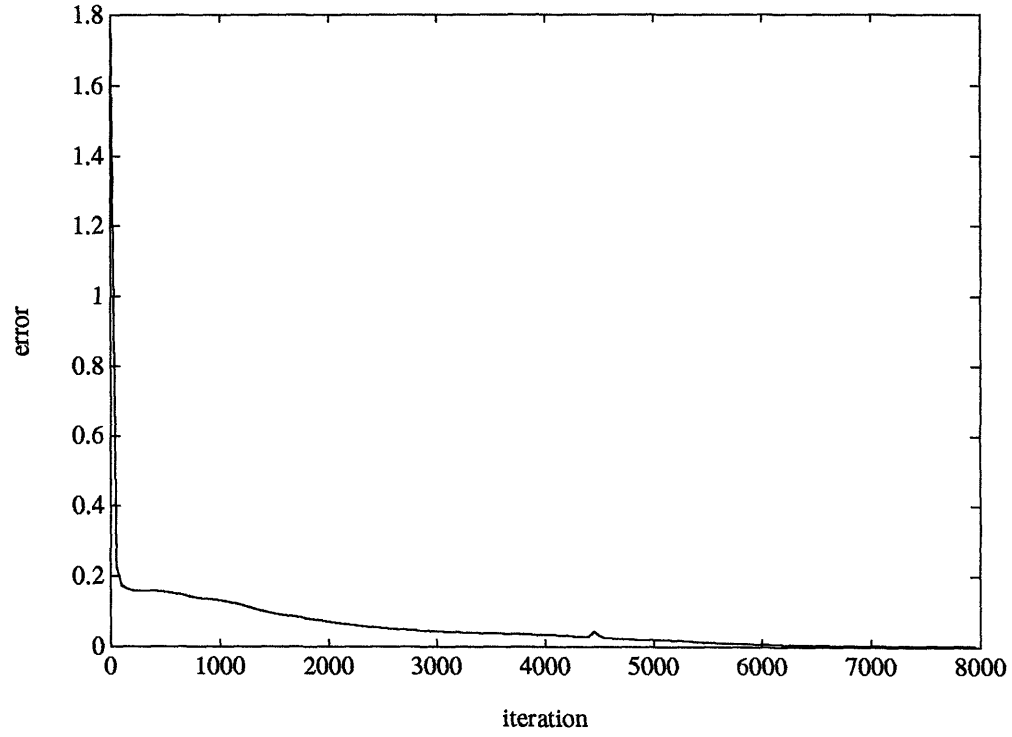
Figures 7.3.1 and 7.3.2 show the approximation of sinusoid functions by neural networks. In Fig. 7.3.1, $y = 0.5 + 0.5\sin(x)$ is used as a nonlinear function and fourteen sample data are given for the approximation. In Fig. 7.3.2, $y = 0.5 + 0.5\sin(x_1 + x_2)$ is used, and thirty-six sample data are given. The neural network has the configuration of 3 layers with 10 cells in the hidden layer. The sigmoid function is used as the cell. We use back-propagation for the identification. Figures 7.3.1 (1) and 7.3.2 (1) show the approximation results. In these figures the broken lines show the results while the parameters are being updated by the back-propagation, and the final approximation is given by the solid line. Figures 7.3.1 (2) and 7.3.2 (2) show the approximation error against the number of iterations.

From the figures we can see the following characteristics of this approximation.

1. The sample data are well interpolated by the neural network.
2. Although it is natural, we can not say anything about how the neural networks behave for the extrapolation outside the region where the sample data are given.
3. The back propagation method requires a fairy large number of iterations. For the given cases, we need at least 5000 iterations. Such large number of iterations comes from the difficulty of the nonlinear minimization problem and the bad convergence of the incremental gradient descent method as discussed in Section 5.4.3.

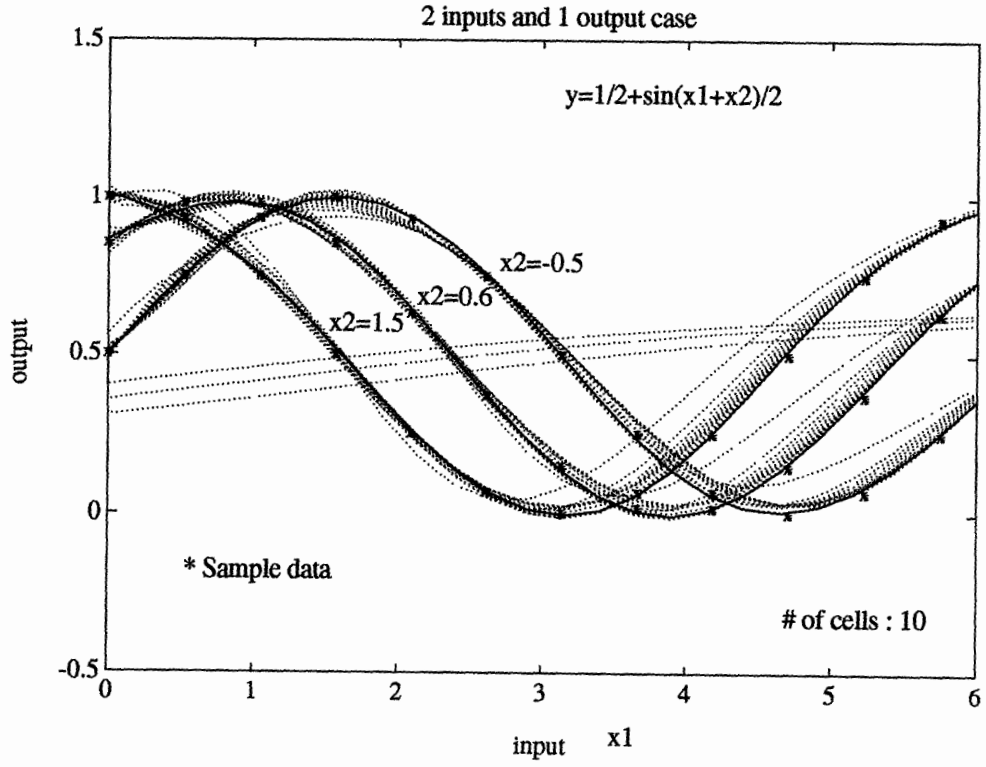


(1) Approximation

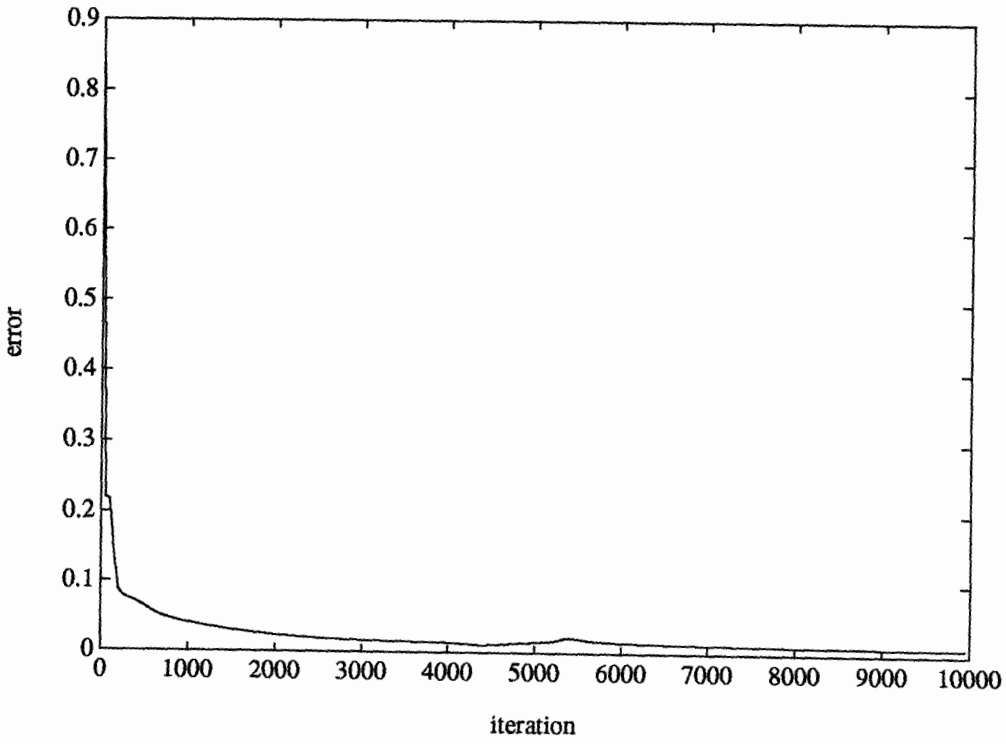


(2) Error

Fig. 7.3.1 Approximation by neural networks (Single input single output case)



(1) Approximation



(2) Error

Fig. 7.3.2 Approximation by neural networks (Two inputs single output case)

7.3.2 Parameter identification of dynamic systems

In this section we discuss how neural networks can identify the unknown dynamics. To simplify the discussion, the plant is assumed to be a simple first order system:

$$G(s) = \frac{1}{s+1} \tag{7.3.1}$$

We take 0.5 [sec] as the sample time. Therefore, the corresponding ARMA model is given by:

$$\begin{aligned} y(t+1) &= \alpha y(t) + (1-\alpha)u(t) \\ \alpha &= \exp(-0.5) \end{aligned} \tag{7.3.2}$$

We assume we know the structure of the plant model. Therefore, the corresponding neural network model is given by

$$y(t+1) = NN(y(t), u(t)) \tag{7.3.3}$$

The series-parallel method by the back-propagation is used for the identification.

(1) The case of the sinusoidal input (Persistently exciting in linear system)

In this case a pure sinusoidal input is given to the plant to identify the parameters of the neural network model. Figure 7.3.3 shows how the identified neural network behaves, responding to the sinusoidal input used for the identification. The neural network is used as a parallel model. In this simulation, the neural network has a structure of 3 layers with 10 cells in the hidden layer. We need more than 10000 iterations to get a satisfactory result. The figure shows that the response by the neural network is almost identical to the response of the actual plant. Note the series-parallel identification succeeds in getting a satisfactory parallel performance for this problem.

However, the response is aggravated when we use the input other than that used for the identification. Figure 7.3.4 shows the response of the neural network when various step inputs are given to the plant. We can see the response differs from the actual plant.

Figure 7.3.5 shows how the neural network has created the mapping from the inputs $y(t)$ and $u(t)$ to the output $y(t+1)$. The mapping should be a flat plan in the three-dimension space, but the neural network creates a winding plane. That is the reason it shows a bad response to the inputs other than that used for the identification.

The simulation shows the following results:

1. The neural network can produce a signal well approximated from the sample signal given by the actual plant.
2. Even if the input is persistently exciting, we can not always get a well-identified model. Consider the above case. The sinusoidal input is persistently exciting for the first order linear system. Therefore if we use a linear model the input is enough to get a satisfactory identification. However, it does not hold for the identification by neural networks.

Considering the characteristics of neural networks, the above discussion is quite natural, because the neural networks work well simply for the interpolation of the data. However, we cannot guarantee anything for the extrapolation as discussed in Section 7.3.1. Therefore, the reasonable selection of a sequence of input is quite important for the identification of the system by neural networks.

However, a question arises: How can we make a reasonable sequence of input? Consider the first order system. The system is given by

$$y(t+1) = f(y(t),u(t))$$

where the function denotes the mapping from the 2-dimensional space of interest Ω_x to the one-dimensional space Ω_y. More precisely the relation can be written as

$$f:\Omega_x \rightarrow \Omega_y \text{ where } \Omega_x \subset R^2 \text{ and } \Omega_x \subset R^1$$

The relation above says a set of the input-output pair $\{y(t),u(t)\}$ should be sufficiently distributed in the region Ω_x. As shown below it is realizable, if the system is of first order. However, if the system is of higher order we can not make such a statement. We can simply say the input given to the system has to be sufficiently rich not only in frequency but also in magnitude. Therefore it requires many experiments to get a satisfactorily identified model.

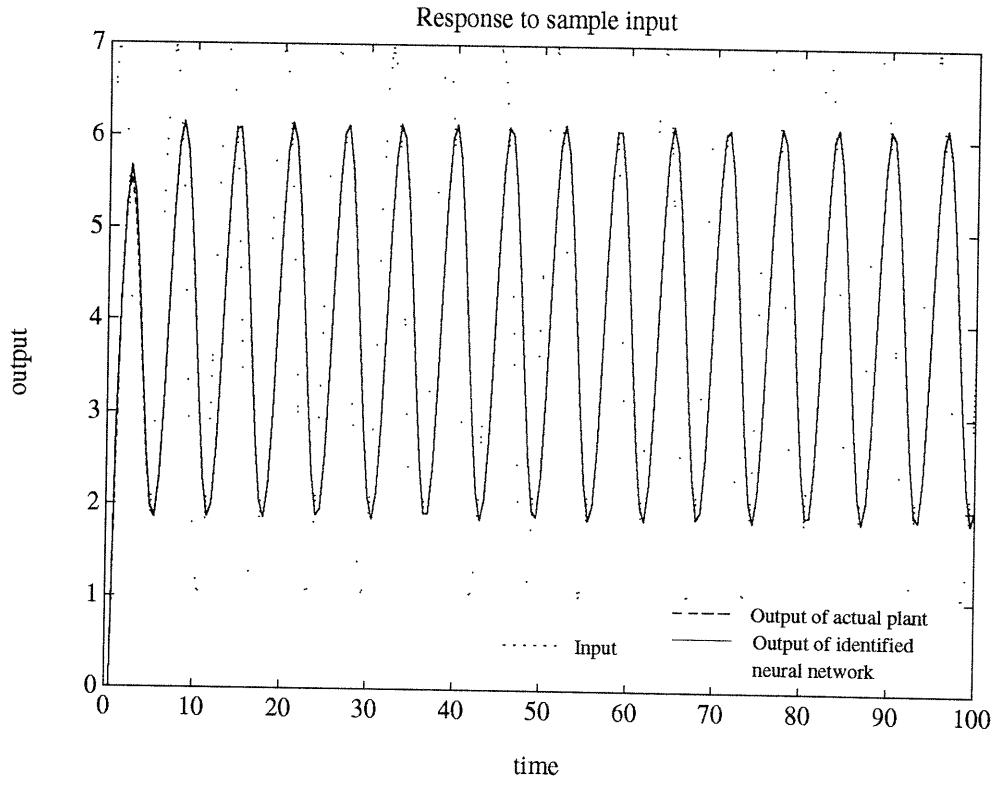


Fig. 7.3.3 Response to pure sinusoidal input

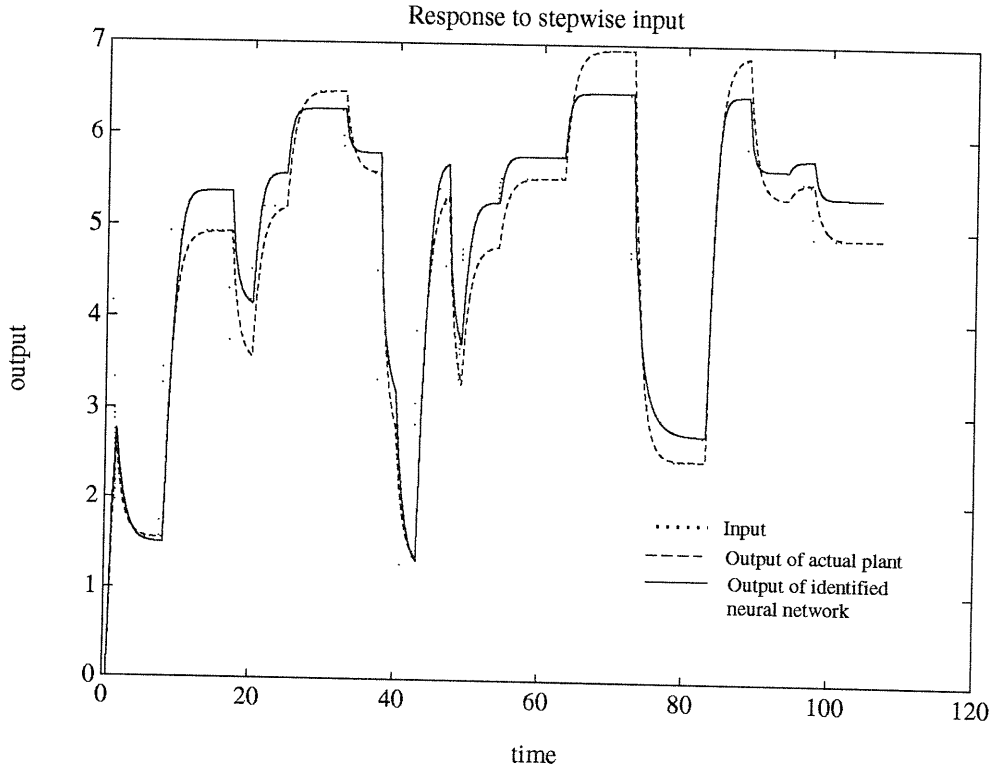
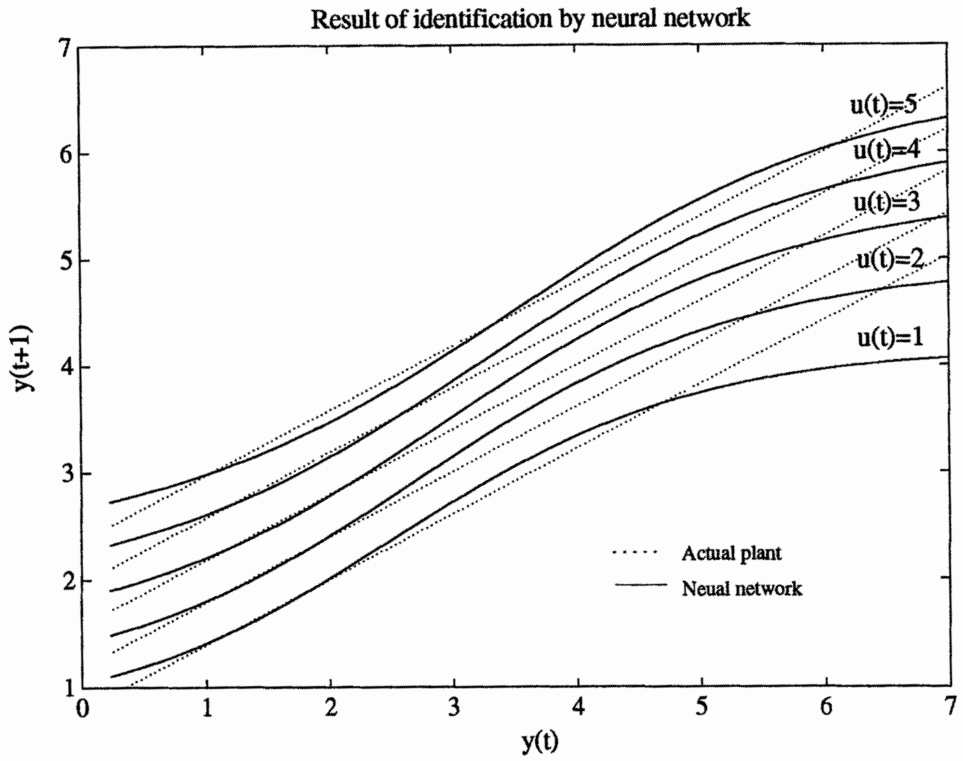


Fig. 7.3.4 Response to step input with various magnitude



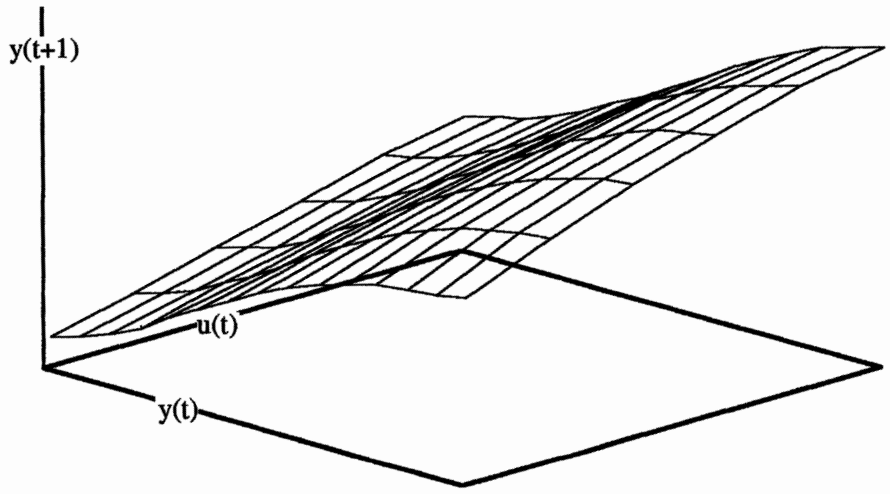


Fig. 7.3.5 Mapping created by neural network

(2) The case of step input with various magnitude

As discussed above, the input given to the plant should be sufficiently rich in frequency and magnitude. If the system is of first order, it is sufficient to give a step input with various magnitudes to the plant as shown in Fig. 7.3.6 to get well-distributed input-output pairs. Fig. 7.3.7 shows the distribution of the sample pairs $\{y(t),u(t)\}$ of the case in Fig. 7.3.6. We can see the sample data are distributed in the region of interest.

Fig. 7.3.9 shows the mapping created by the neural network. The mapping almost meets with the actual plant. Thus the neural network produces a satisfactory response for the input other than that used for the identification, as shown in Fig. 7.3.8, where we can see a good agreement between the response of the neural network and that of the actual plant for the sinusoidal input with some frequencies.

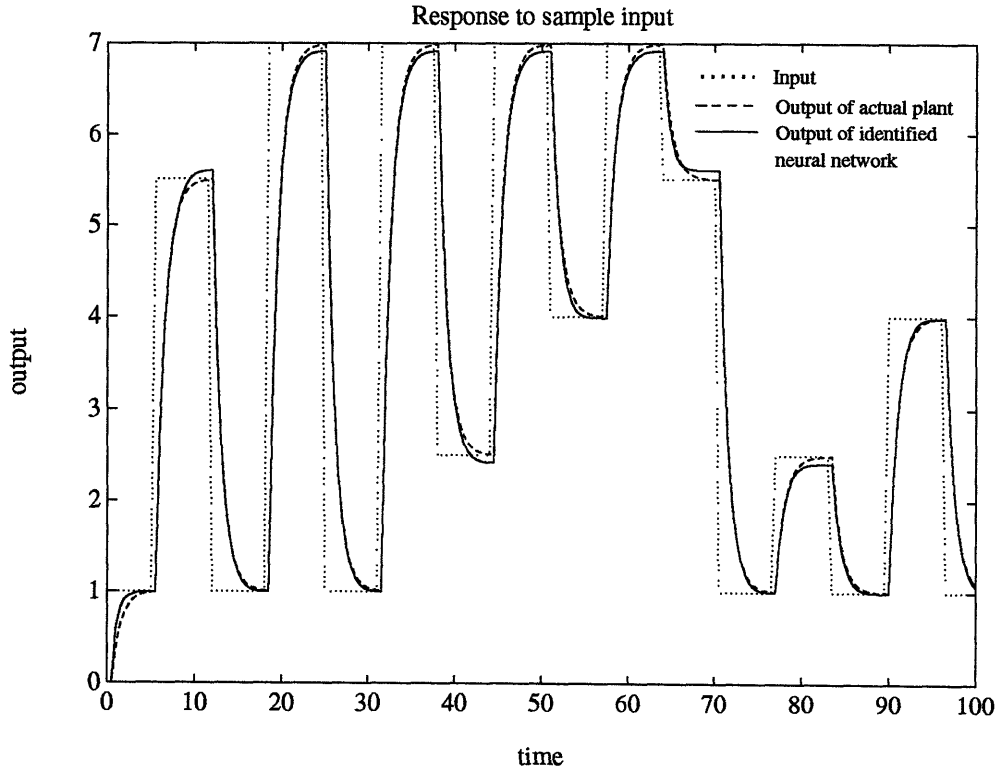


Fig. 7.3.6 Response to step input with various magnitude

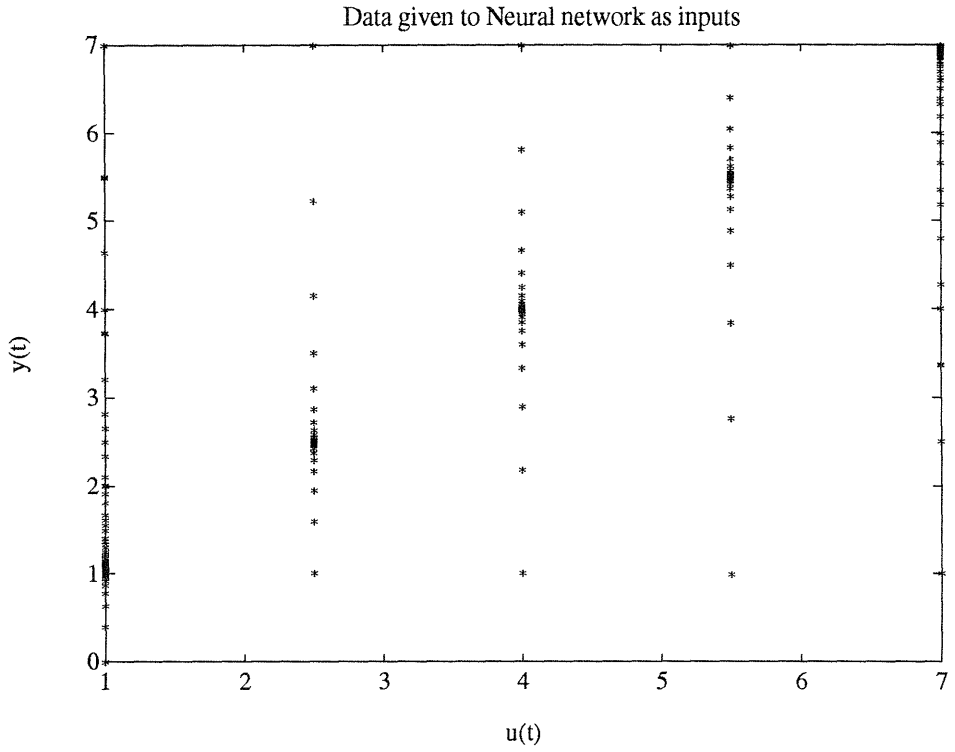


Fig. 7.3.7 Distribution of input-output pairs

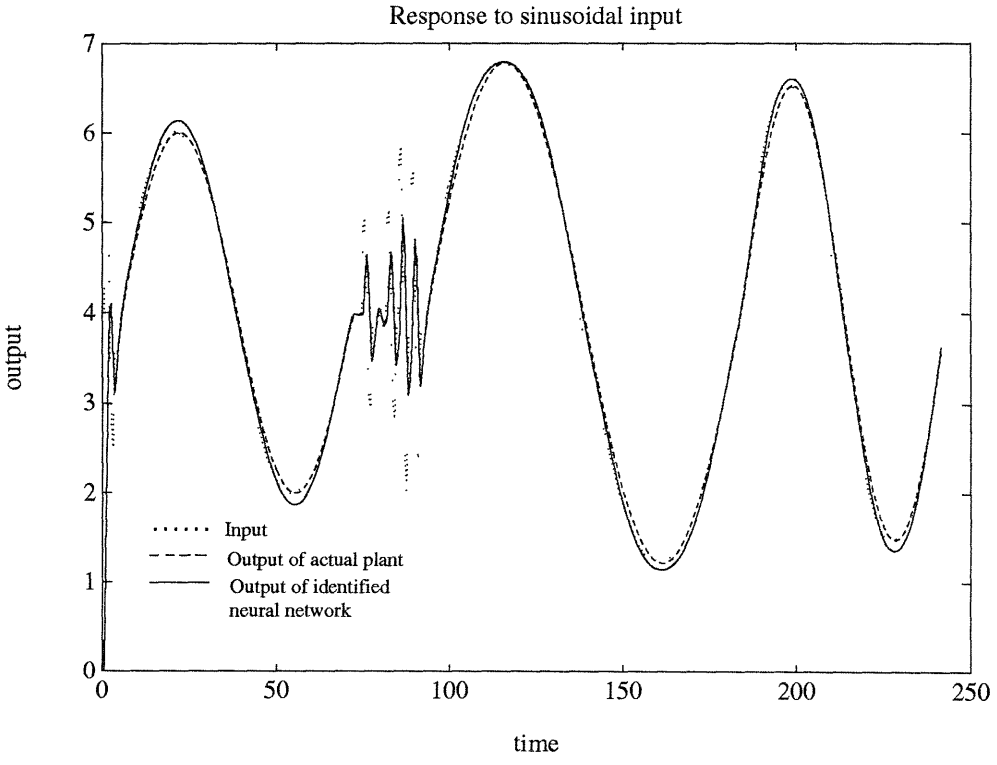


Fig. 7.3.8 Response to sinusoidal input with various frequencies

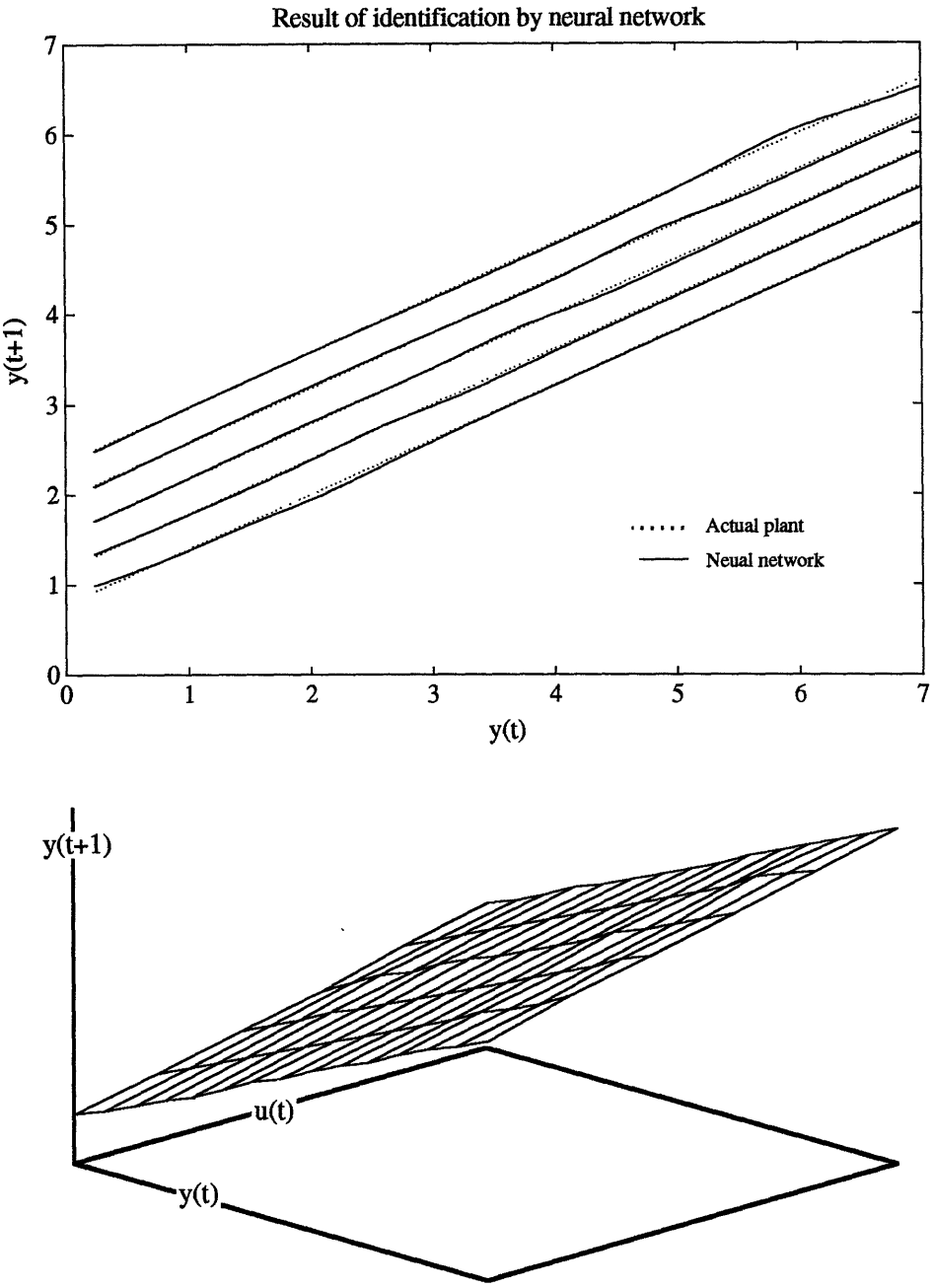


Fig. 7.3.9 Mapping created by neural network

7.3.3 Concluding remarks

The simulation above for the linear first order system shows the following results, which would yield some problems for the actual implementation.

1. Back propagation requires a quite large number of iterations.
2. The interpolation of the data is well established by neural networks. However, we cannot guarantee anything about the extrapolation. Therefore, if the region of interest is larger than the region where the sample data are given, the identified neural network may produce a wrong output.
3. The input given to the plant for the identification should be rich with respect to frequency and magnitude. However, there is no theory for how we can construct such an input signal.
4. The discussion **3** implies that the identification requires a large number of experiments.
5. If the plant is of first order, we have only to give a step input with various magnitudes to achieve the signal discussed in **3**.
6. The visualization as shown in Figs. 7.3.5 and 7.3.9 is helpful for understanding the nonlinear behavior of the system.
7. If the plant is of higher order, we may have to determine the input signal by trial and error. Furthermore, we have no way to ensure how the neural network behaves except through the repetition of laborious simulations because the neural network just creates a mapping from the inputs to the outputs. The values of parameters do not give us any suggestion about the properties of the system unlike the case of linear systems. The visualization shown in Fig. 7.3.5 and 7.3.9 may not always be applicable.

7.4 Selection of control using neural networks

Some nonlinear control methods were discussed in Chapter 5. It is crucial to select the most feasible method, considering the drawbacks of neural networks. In this section we discuss the control scheme feasible for our problem.

7.4.1 Feed-back control with fixed parameters or adaptive control?

As discussed in Section 5.1.4, we must consider two types of control, feed-back control with fixed parameters and adaptive control, as the feasible control schemes using neural networks.

First let us consider adaptive control. All methods discussed in Section 5.7.2 use a back-propagation method (an incremental gradient descent method) as a parameter estimator. However, as mentioned in Section 5.4.3 and Section 7.3.3, the incremental gradient descent method has a fundamental drawback: quite slow convergence. The convergence is 100 times or more slower than that of linear estimations such as a recursive least square algorithm. (See the figures in Section 7.3.1, and the papers by Narendra (1990, 1991) and Chen (1990). It requires ten thousands of iterations or more to get a satisfactory result.) Obviously controller adaptation strongly depends on how fast the controller estimates the parameters. Therefore, it seems to be almost impossible to apply the adaptive scheme to the control of the actual plant from a practical view point.

From the discussion above, we can not but use a nonlinear feed-back control with fixed parameters. In general the design of the controller with fixed parameters is simpler than that of the adaptive scheme. However, we still raise many questions:

(1) How can we design the controller using neural networks?

(2) How can the stability be guaranteed?

(3) How is the robustness?

and so on.

In this thesis we will focus on the design method, and just show how the nonlinear controller using neural networks performs for the control of the cooling rate.

7.4.2 Model structure

As discussed in Section 5.3.2, Narendra et al. proposed four types of model structures. We have to choose the suitable one. However, we have no clue for the choice because we do not know clearly what properties each model structure has.

As seen in Section 5.6, almost all nonlinear control methods are based on the cancellation of the nonlinearity of the plant dynamics. Therefore, a better model necessarily shows better performance because we will use the scheme of feed-back control with fixed parameters. Therefore, we have no way other than to choose the most general structure, **model 4**:

$$y(t+1) = NN(y(t), y(t-1), \cdots, u(t), u(t-1), \cdots) \qquad (7.4.1.a)$$

If we take an adaptive scheme, we may be able to use a structure which we can treat more easily such as **model 1** and **model 2**:

$$y(t+1) = \sum_{i=0}^{n-1} \alpha_i y(t-i) + NN[u(t), u(t-1), \cdots] \qquad \textbf{(model 1)} \qquad (7.4.1.b)$$

$$y(t+1) = NN[y(t), y(t-1), \cdots] + \sum_{i=0}^{m-1} \beta_i u(t-i) \qquad \textbf{(model 2)} \qquad (7.4.1.c)$$

7.4.3 Order of model

Obviously the model should accommodate all behavior of the plant. However, as discussed in Section 7.3, the increase of the order of the nonlinear model leads to the complex design of the experiments for the identification. Simple observation of the cooling rate response shows that a first order model seems to be enough to accommodate and approximate the dynamics, ignoring the effect of the time delay. Therefore, we will use a first order model as the delay-free part of the cooling rate dynamics.

The discussion above does not always hold for other systems. Therefore, control using neural networks would have severe limitations until we establish the design method of the experiments for the nonlinear identification.

7.4.4 Treatment of time-delay

Two types of control methods, Smith's method and one-step-ahead predictive control, were discussed in Section 5.5. The predictive control was used as a linear adaptive control law in Chapter 6.

First let us discuss the applicability of the predictive control. Consider a nonlinear time-delay system whose delay-free part is of first order. Then, the ARMA model is given by

$$y(t+1) = NN_p(y(t), u(t-d+1)) \tag{7.4.2}$$

The predictor form can be written as:

$$\begin{aligned} y(t+d) &= NN_p(y(t+d-1), u(t)) \\ &= NN_p(NN_p(y(t+d-2), u(t-1), u(t)) \\ &\quad \vdots \\ &= \overline{NN_p}(y(t), u(t), u(t-1), \cdots, u(t-d+1)) \end{aligned} \tag{7.4.3}$$

The predictive control is to look for the function $NN_c()$ such that

$$\begin{aligned} y_d(t+d) &= \overline{NN_p}(y(t), u(t), u(t-1), \cdots, u(t-d+1)) \\ u(t) &= NN_c(y(t), u(t-1), \cdots, u(t-d+1), y_d(t+d)) \end{aligned} \tag{7.4.4}$$

Therefore, we encounter again the problem of designing the nonlinear function $NN_c()$ which has more arguments than two. The design would be difficult for the same reason as nonlinear identification of a higher order system. Therefore, the concept of predictive control:

1. Identify the unknown parameters in a predictor form.

2. Design the control based on the predictive form.

could not be applied even if the delay free part of the system is of first order.

To avoid the complexity of the design, we had better use Smith's method, because we can ignore the time-delay and design the controller simply for the delay-free part. The problems for Smith's principle are

1. We may not use a series-parallel identification as discussed in Section 7.2.

2. Therefore, we may not use a back-propagation method for the identification.

The control system based on Smith's principle has a structure shown in Fig. 7.4.1.

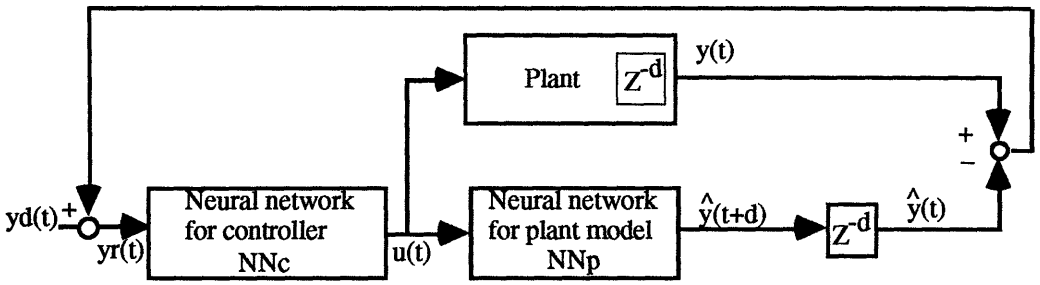


Fig. 7.4.1 Control based on Smith's principle

7.4.5 Control based on Smith's principle

(1) Feasible control law in inner loop

The control in the inner loop is a nonlinear feed-forward control based on an identified nonlinear ARMA model. The following control methods can be considered, based on the discussion in Section 5.6. The design of the controller is to look for such a neural network NN_c as is given below.

Predictive control

$$NN_p(y(t), NN_c(y(t), y_d(t))) = y_d(t) \tag{7.4.5}$$

Feed-back linearization

$$NN_p(y(t), NN_c(y(t), y_d(t))) = y_d(t) - k(y_d(t) - y(t)) \tag{7.4.6}$$

Weighted predictive control

$u(t) = NN_c(y(t), y_d(t))$ such that minimizes

$$E = (NN_p(y(t), u(t)) - y_d(t))^2 + \lambda u(t)^2 \tag{7.4.7}$$

Generic model control

$$NN_p(y(t), NN_c(y(t), y_d(t))) = k_p(y_d(t) - y(t)) + k_i \sum_{\tau}^{t} (y_d(\tau) - y(\tau)) \tag{7.4.8}$$

We can use the back-propagation method with a distal system for generating the neural network $NN_c()$. The further discussion will be found in Section 7.6.

(2) Other issues associated with Smith's principle

Outer loop

The inner loop has no feed-back. Therefore, all the plant deviation from the model is accommodated by the outer loop. The deviation will be caused by some factors such as a nominal model error, a plant perturbation, and a disturbance.

The outer loop seems to be too simple to accommodate the above deviation, as shown in Fig. 7.4.1. Therefore, we had better add some functions to this feed-back loop. However, the modification is quite difficult because both the plant and the controller are nonlinear, and there are no techniques to analyze the problem.

We can easily see this configuration can not eliminate a steady-state error because there is no integral effect in the control. Therefore it may be better to add at least an integrator to this loop.

Varying time delay

If the time-delay in the model has an error, the performance of the control will be degraded. Therefore we have to know the exact value of the time-delay. However, the time-delay of the cooling rate varies by the input and the state of the plant. Therefore, we must estimate it on-line. Unfortunately we have no method to estimate the time-delay in this way.

7.5 Identification by neural networks for cooling rate

In this section we discuss the nonlinear identification by neural networks for the cooling rate. The discussion is held by the simulation based on the analytical model. In accordance with the discussion in Section 7.4 we use a nonlinear first order system with a time-delay as the model describing the dynamics of the cooling rate.

7.5.1 Input given for the identification

Figure 7.5.1 shows the travel speed given for the identification of the cooling rate dynamics. The simulation result of the response of the cooling rate is also shown in the figure. The condition for the simulation is same as the previous chapters. The travel speed is selected from the following points based on the discussion in Section 7.2 and 7.3.

1. Since we assume the system is a nonlinear first order system, the step input with various magnitudes is given for the identification. The step input is also beneficial to observe the time-delay clearly. If we use other inputs, we can not see the effect of the time delay precisely due to the phase shift of the system.
2. Since we use Smith's principle for the control, we have to identify the dynamics of the delay-free part. Therefore, after we get the response, the data of the input signals are shifted so that there seems to be no time-delay. The shifted travel speed is plotted in Fig 7.5.1.
3. The time-delay is dependent on the input. To deal with the varying-time delay, first the time-delay that takes place every step input is measured, and then each step input data is shifted by the value corresponding to the measured time-delay.

In the method above, we assume that the time-delay is determined by the initial value and the final value in the step-wise change of the input. This assumption is not necessarily true, but there is no way to measure the exact time-delay in the transient response.

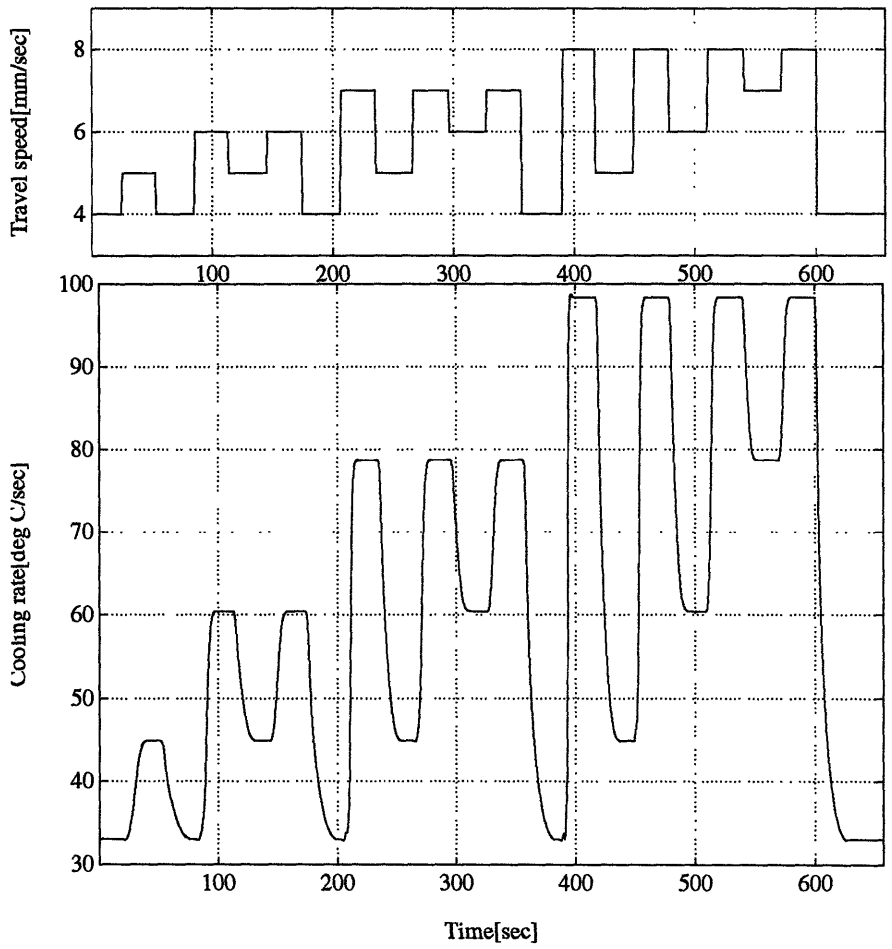


Fig. 7.5.1 Sample data for the identification by neural networks

7.5.2 Result of series-parallel identification

Fig. 7.5.2 shows the result of the series-parallel identification using the sample data shown in Fig. 7.5.1. In the identification the cooling rate is scaled in the same manner as adaptive control as shown in (6.2.10). We use a nonlinear model of first order, following the discussion in Section 7.3. The neural network has 10 cells in the hidden units. The identification is made by the back-propagation method.

In Fig. 7.5.2 the actual plant output, the series-parallel estimate, and the parallel estimate are shown. The series-parallel estimate completely follows the actual plant output. Therefore the neural network has a capacity to make a quite good one-step ahead prediction. However the performance of the parallel estimate is not always satisfactory as seen in the figure. This result comes from a presence of the model error, as discussed in

Section 7.2. The neural network obtained by the series-parallel identification, therefore, can not be used for Smith's principle, because Smith's principle requires good performance for the parallel estimate.

The identification by back-propagation converges to a minimum with about 300000 iterations. A very small step size was required to get a reasonable result.

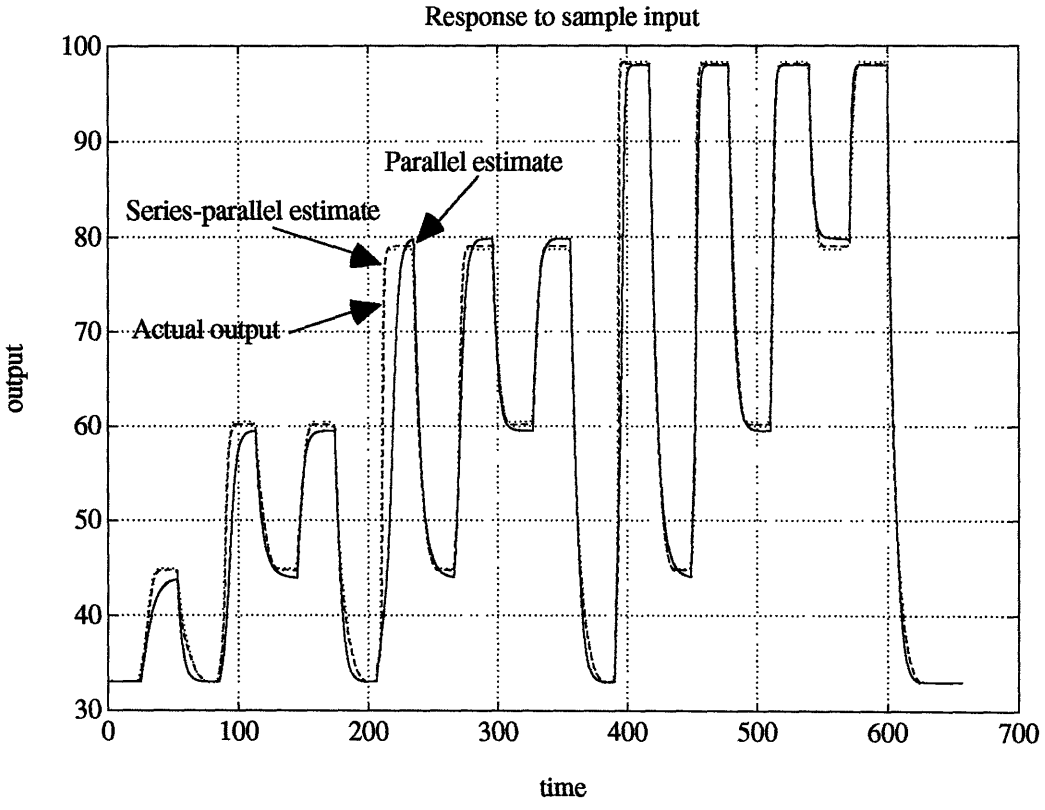


Fig. 7.5.2 Series-parallel identification

7.5.3 Result of parallel identification

As shown in Section 7.5.2, the series-parallel identification does not produce a satisfactory result of the parallel estimate $\hat{y}(t)$.

In this section we use the parallel identification based on the conjugate gradient algorithm discussed in section 7.2.3. Figure 7.5.3 shows the result by the parallel identification. Good performance is observed, although there is a small error.

The conjugate gradient method also required a quite large number of iterations due to redundant but necessary computations in the line minimization. (In the algorithm, the line minimization is executed by the algorithm of Brent's method. This method is quite reasonable and considered to be the most efficient algorithm. Even if so, it sometimes shows unavoidable redundancy.) In the case of Fig. 7.5.3 we needed 200 iterations, which corresponds to 120000 iterations in the back-propagation method.

Although the conjugate gradient method needs a quite large number of iterations, we obtained a satisfactory performance, which has never achieved by the series-parallel identification. We shall use this neural network as the plant model. Figure 7.5.4 shows the obtained map $f:\{y(t),u(t)\} \rightarrow y(t+1)$. We cannot explain anything about why we get such a mapping. However, we can see the model must be stable by simply observing how the line $y(t+1) = f(y(t),u(t))$ with a constant $u(t)$ crosses the equilibrium line $y(t+1) = y(t)$.

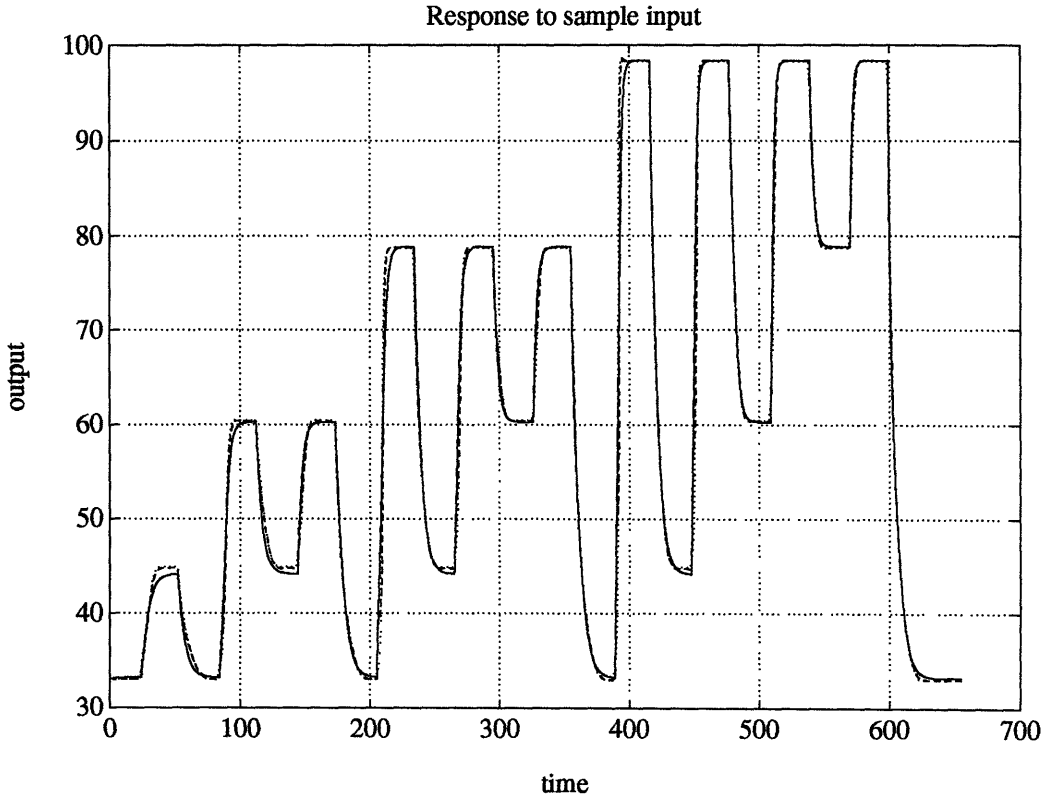


Fig. 7.5.3 Parallel identification

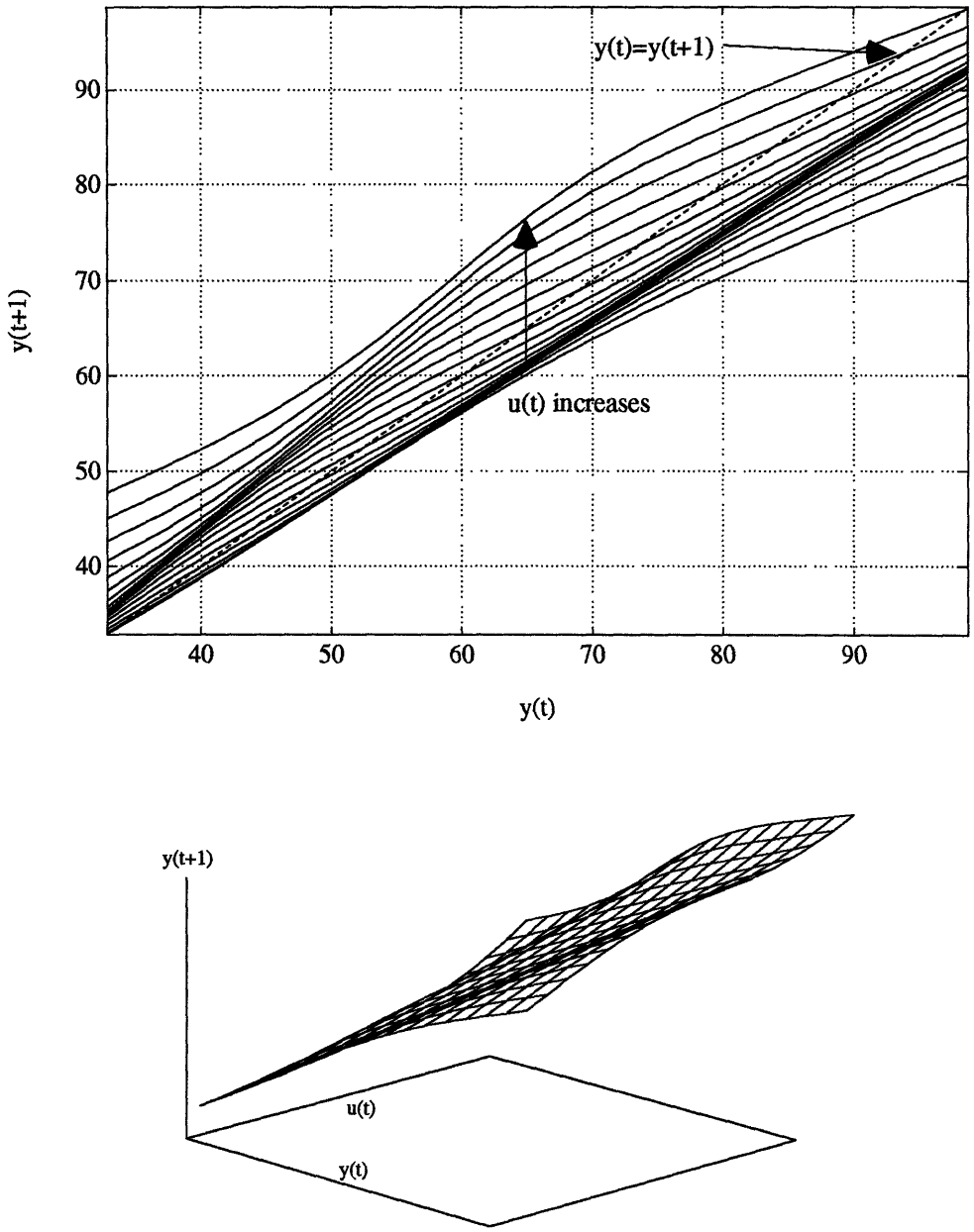


Fig. 7.5.4 Mapping $y(t+1) = NN(y(t), u(t))$

7.6 Control using neural networks for cooling rate

We use Smith's principle as the control of the cooling rate in accordance with the discussion in Section 7.4. In this section we discuss how we can design the control based on Smith's principle for the cooling rate.

7.6.1 Selection of control law for inner loop

Some control structures were presented in Section 7.4.3. Let us choose a control law that is suitable for the inner loop of the control based on Smith's principle.

First, consider weighted predictive control. The control is tuned by the coefficient λ. Therefore, we always have to look for a neural network every time the coefficient λ is changed. Since it usually needs a large number of computations to organize neural networks, the design by the weighted one-step-ahead control is quite laborious. Generic model control includes an integral. There is no reason we put an integrator in this feed-forward control. Therefore, we pay attention to feed-back linearization and predictive control. As seen easily, the predictive control is essentially equivalent to the feed-back linearization with the coefficient zero. It must be also noted that the design is to find the nonlinear function $NN_c()$ such that $NN_p(y(t), NN_c(y(t), h)) = h$. Then, if the input given to the plant is determined by

$$u(t) = NN_c(y(t), y_d(t)) \tag{7.6.1}$$

the control is predictive control.
If the input given to the plant is determined by

$$u(t) = NN_c(y(t), y_d(t+1) + k(y(t) - y_d(t)) \tag{7.6.2}$$

the control becomes a feed-back linearization method.

In order to find the nonlinear function $NN_c()$ such that $NN_p(y(t), NN_c(y(t), h)) = h$, we use the back-propagation with a distal system which minimizes the cost function:

$$E = \sum_{k=1} (h_k - NN_p(y_k, NN_c(y_k, h_k)))^2 \qquad \{y_k, h_k\}_k \text{ are a set of sample pairs} \tag{7.6.3}$$

7.6.2 Back-propagation with a distal system

Figure 7.6.1 shows the nonlinear controller obtained by minimizing the cost function (7.6.3). The figure plots the approximated value of the travel speed necessary to change the cooling rate from the present value $y(t)$ to the target value h.

Due to the nonlinearity of the plant, the controller $NN_c()$ also shows a nonlinearity. A neural network which has 40 cells in the hidden layer is used for $NN_c()$. 900 sample data which comprise 30x30 meshed sample pair $\{y(t),h\}$ ranging from 30 to 100 are given. The range is determined by the possible value of the cooling rate. The following techniques are necessary to obtain a satisfactory result.

1. The neural network $NN_p()$ describing the plant model is determined by the sample data in a boundary region. Therefore, we do not know how the neural network behaves outside the region. Consider the design of the controller $NN_c()$, which simply minimizes the cost function:

$$E = \sum_{k=1} (h_k - NN_p(y_k, NN_c(y_k, h_k)))^2$$

We can easily see that the design does not take into consideration the presence of the boundary associated with $NN_p()$. Therefore, there is a big possibility that the cost function is minimized outside the region. Then the generated neural network would not show any reasonable result.

To avoid such bad generation of the function $NN_c()$, we have to impose a constraint on the minimization. It is achieved by simply giving the following additional cost to the cost function:

$$E = \sum_{k=1} (h_k - NN_p(y_k, NN_c(y_k, h_k)))^2 + c$$

$$c = \begin{cases} a(NN_c(y_k, h_k) - b_l)^2 & (NN_c(y_k, h_k) < b_l) \\ a(NN_c(y_k, h_k) - b_u)^2 & (NN_c(y_k, h_k) > b_u) \\ 0 & (Otherwise) \end{cases} \qquad (7.6.4)$$

where b_l denotes the lower bound, and b_u denotes the upper bound. The cost function gives a penalty if the output of $NN_c()$ is deviated from the desirable

range. By this penalty the minimization outside the specified region can be avoided.

2. If the model error takes place at the sample pair where $y(t) = h$, it results in the stead-state error. We would like to reduce a steady-state error as much as possible. It is better to give a big penalty when $y(t) = h$ by using the cost function:

$$E = \sum_{k=1} d(h_k - NN_p(y_k, NN_c(y_k, h_k)))^2 + c$$
$$d = \begin{cases} d_1 (>1) & (y_k = h_k) \\ 1 & (y_k \neq h_k) \end{cases} \tag{7.6.5}$$

We take 3 mm/sec and 9 mm/sec as the lower bound and the upper bound in (7.6.4). We also take 1.0 and 10.0 as the coefficient a in (7.6.4) and d_1 in (7.6.5) respectively.

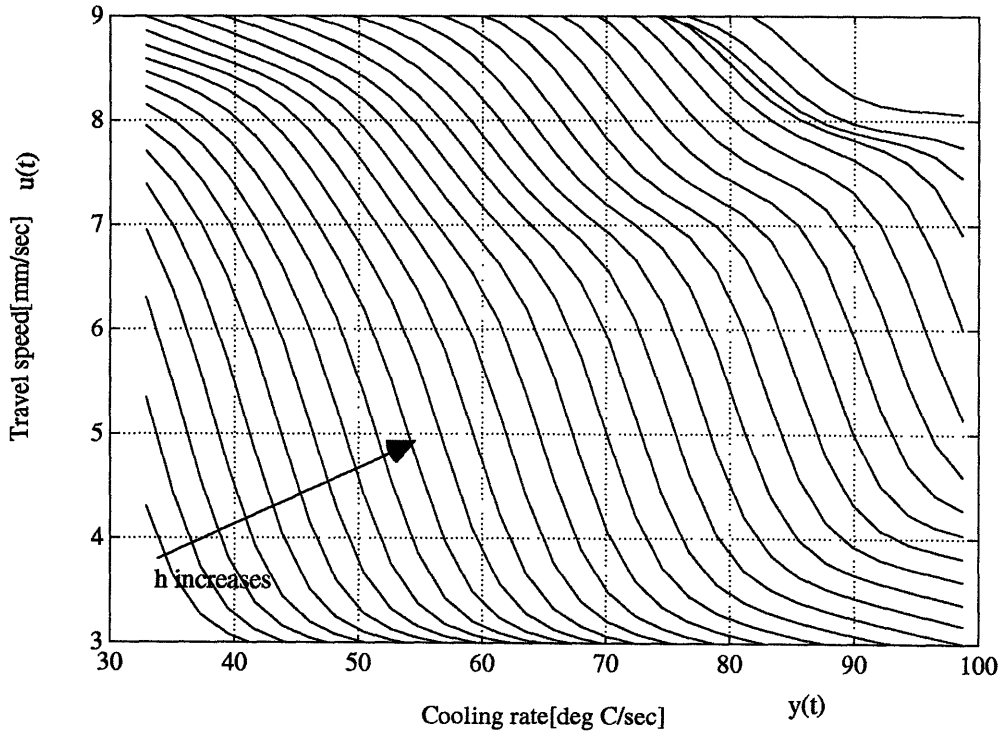
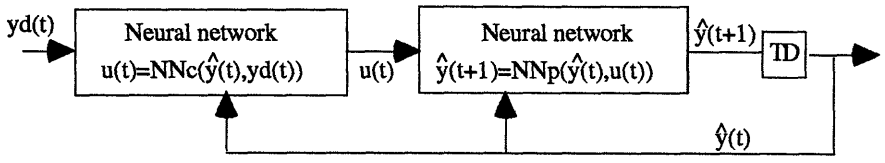


Fig. 7.6.1 Nonlinear controller $u(t) = NN_c(y(t), h)$

Figure 7.6.2 shows how the two networks respond to the reference input, when we combine $NN_c()$ with $NN_p()$. In the figure, the broken line shows the reference input, and the solid line shows the response of the combined neural networks. The control signal is determined by $u(t) = NN_c(y(t), y_d(t))$, where the input signal $u(t)$ is limited within the range from 4 mm/sec to 8 mm/sec. The corresponding control is a predictive control. The figure shows a quite satisfactory response. That verifies the controller which given in Fig. 7.6.1 will bring about a reasonable control.



(Block diagram)

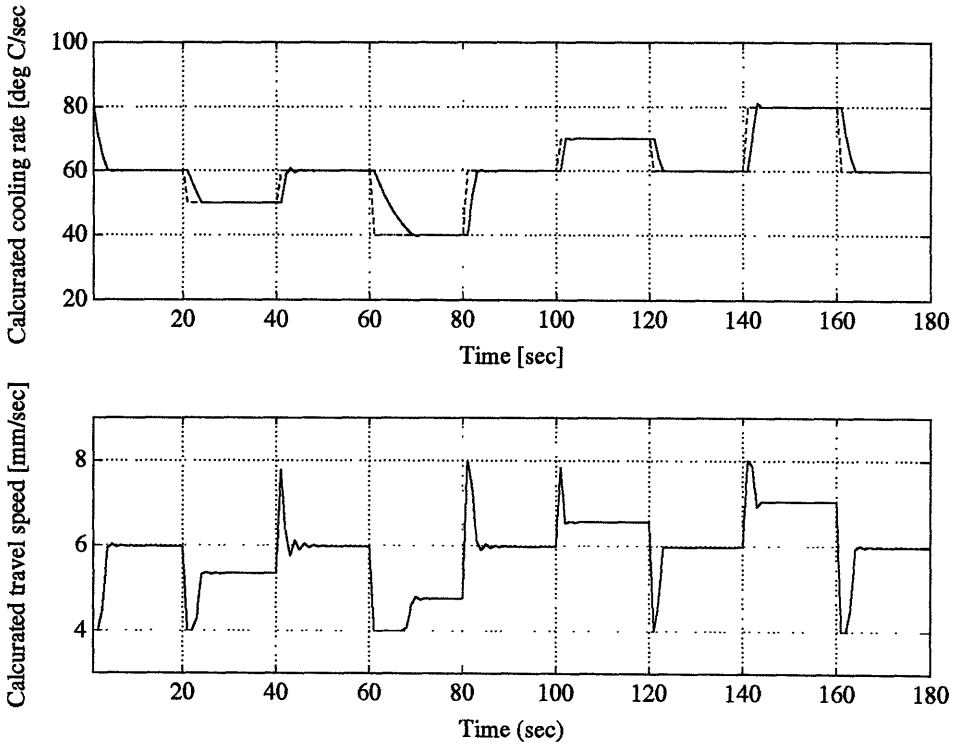


Fig. 7.6.2 Response of the combined neural networks

7.6.3 Performance of inner loop control

Figures 7.6.4 and 7.6.5 show how the nonlinear controller for the delay-free part (the inner loop control) performs for the actual plant. The block diagram of the system is shown in Fig. 7.6.3. In Fig. 7.6.4 the controller has the form:

$$\begin{aligned} u(t) &= NN_c(\hat{y}(t+d-1), y_d(t)) \\ \hat{y}(t+d) &= NN_p(\hat{y}(t+d-1), u(t)) \end{aligned} \qquad \text{(Predictive control)} \qquad (7.6.6)$$

while in Fig. 7.6.5 it has the form:

$$\begin{aligned} u(t) &= NN_c(\hat{y}(t+d-1), h) \\ h &= y_d(t) + k(\hat{y}(t+d-1) - y_d(t)) \\ \hat{y}(t+d) &= NN_p(y(t+d-1), u(t)) \\ k &= 0.5 \end{aligned} \qquad \text{(Feed-back linearization)} \qquad (7.6.7)$$

In the equation above, we add a time-delay term d to see the difference from the actual plant dynamics which would be given by the form:

$$y(t+1) = f(y(t), u(t-d+1)), \text{ or } y(t+d) = f(y(t+d-1), u(t))$$

Note that $\hat{y}(t+d)$ denotes the prediction of $y(t+d)$ which is computed at time t. In the figures the solid line shows the actual plant response and the broken lines show the values of $y_d(t)$ and $\hat{y}(t+d)$.

Some steady-state errors and small overshoots take place due to the model error. However, the over all response is quite satisfactory. In Fig. 7.6.5 we can see the plot is like the response of a first order linear system by the effect of the feed-back linearization. We also see the neural network $NN_p()$ makes a satisfactory prediction $\hat{y}(t+d)$

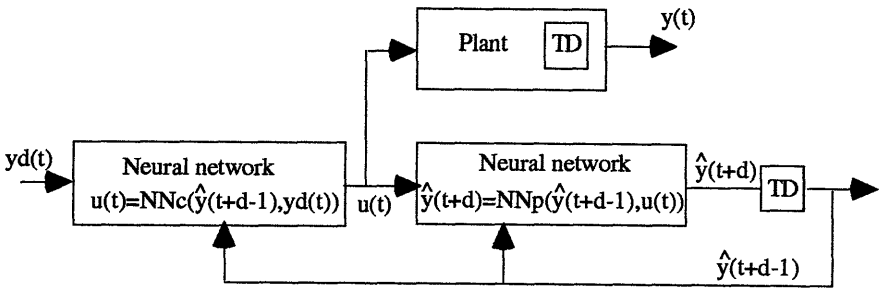


Fig. 7.6.3 Block diagram of the inner loop control

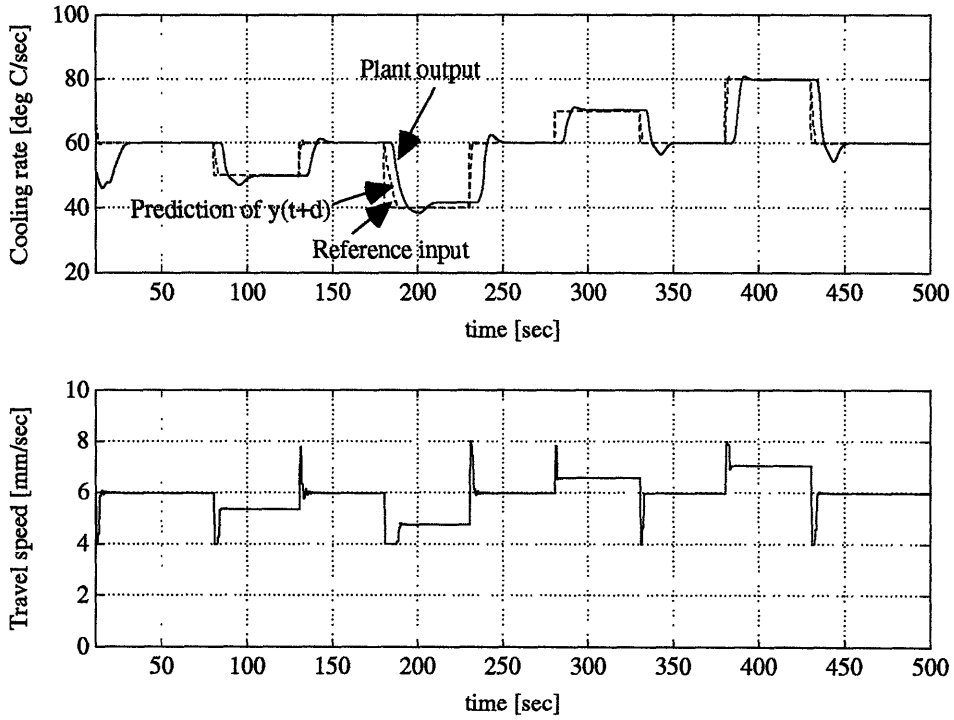


Fig. 7.6.4 Predictive control

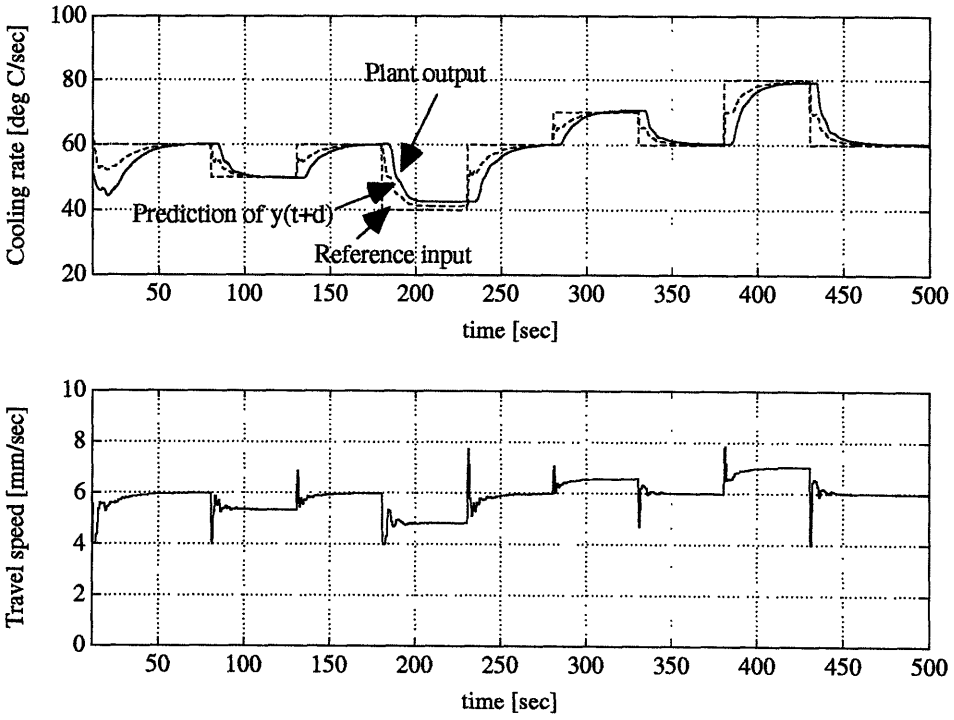


Fig. 7.6.5 Feed-back linearization (k=0.5)

7.6.4 Outer loop control

The control discussed in Section 7.6.3 is a feed-forward control. Therefore we need to add a feed-back loop to accommodate a disturbance and a perturbation of plant dynamics from the model. In Smith's principle, the error between the model output and the plant output is simply added onto the reference input as shown Fig. 7.6.6. In this section we discuss some problems which take place when we apply the Smith's principle to the cooling rate control.

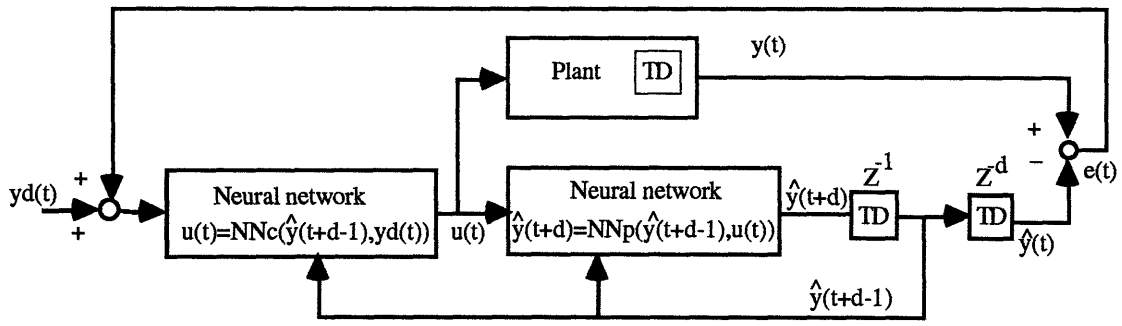


Fig. 7.6.6 Block diagram of the control based on Smith's principle

Treatment of a varying time-delay

In Smith's principle, the time-delay should be known so that we can establish the feed-back of the error. However, the time-delay of the cooling rate varies, depending on the input and the state of the system. Therefore, we have to implement an algorithm which estimates the time-delay on-line. Since we have no theoretical method to deal with such problem, we will use the following simple algorithm:

$$\hat{d} = t - \tau \qquad \tau = \underset{\tau_l \le \varsigma \le \tau_u}{\arg} \min(y(t) - \hat{y}(\varsigma))^2 \tag{7.6.8}$$

A series of estimate $\{\hat{y}(t'), \hat{y}(t'-1), \hat{y}(t'-2), \cdots\}$ is available in Smith's principle where $t' = t + d$ and the time-delay d is unknown. The idea to estimate the time-delay is simply to look for the estimate $\hat{y}(\tau)$ whose absolute error from the actual measurement $y(t)$ is minimum, then determine the time-delay. Since the time-delay seems to vary from 2[sec] to 6[sec] as we discussed in Chapter 4, the bound of τ is given by $\tau_l = t - 3$ and $\tau_u = t - 7$.

Figure 7.6.7 shows the response of the control by Smith's principle using a time-delay estimate. In this simulation, 0.2 is taken as the gain of the feed-back linearization k. The figure shows that the input signal becomes noisy due to the effect of the feed-back loop, compared with the pure feed-forward control shown in Fig. 7.6.6. Next we can see the steady state error still remains, simply because the Smith's principle is just concerned with the error between the plant output and the model output, and does not care about the error between the plant output and the reference input. To see the problems above more clearly, the case where the constant disturbance 10 °c/sec is added to the output of the plant is shown in Fig. 7.6.8. We can see the disturbance is reduced by the control, but is not completely eliminated. We also see the input signal is rather fluctuated due to the bad estimate of the time-delay.

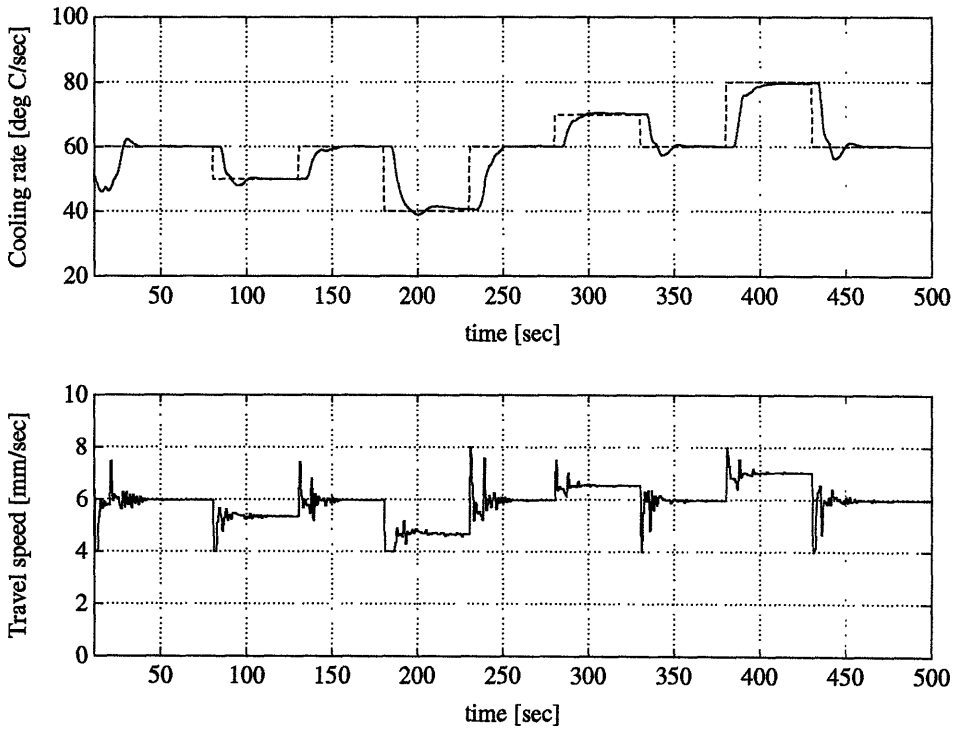


Fig. 7.6.7 Feed-back control by Smith's principle

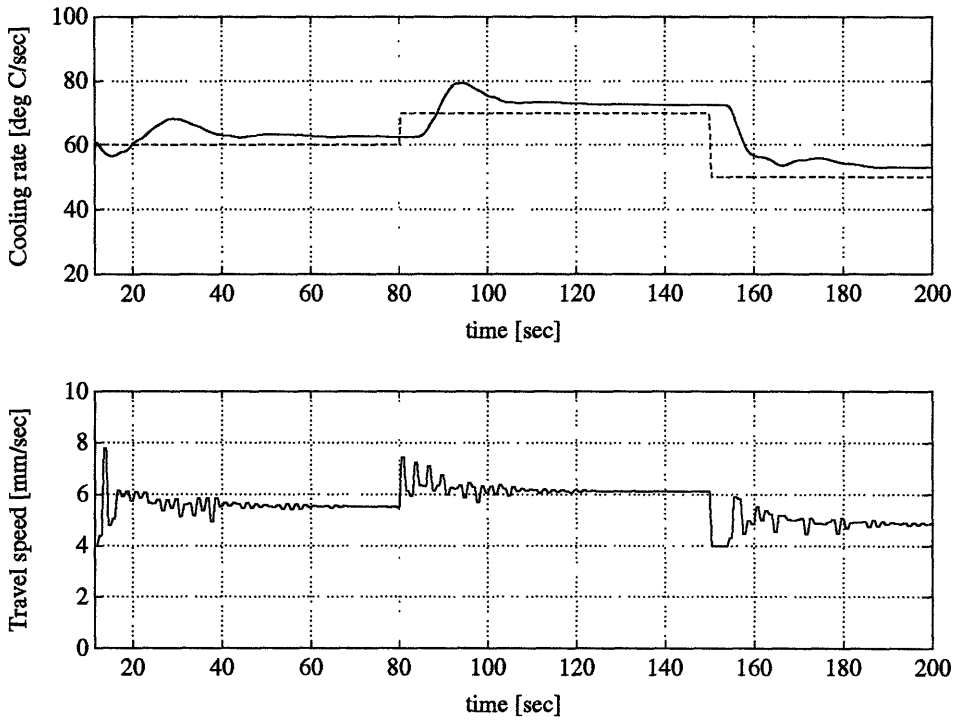


Fig. 7.6.8 Feed-back control by Smith's principle for a disturbance of 10 °c/sec

Filter of error estimate and integrator

A filtering of an error estimate and an integrator are introduce to deal with problems, a fluctuating input signal and a steady-state error. The error estimate $\hat{e}(t)$ is given by

$$\hat{e}(t) = y(t) - \hat{y}(t' - \hat{d}) \tag{7.6.9}$$

from (7.6.8).
Then, it is filtered out by a first order filter:

$$\tilde{e}(t) = k_f \hat{e}(t) + (1 - k_f)\tilde{e}(t) \tag{7.6.10}$$

That simply makes an exponentially weighted smoothing to reduce a fluctuation of the error estimate.

Next, let us consider an integrator to eliminate a steady-state error. The integrator $\Delta y_I(t)$ is given by:

$$\Delta y_I(t+1) = \Delta y_I(t) + k_i(y_d(t+d) - \tilde{y}(t+d))$$
$$\tilde{y}(t+d) = \hat{y}(t+d) + \tilde{e}(t)$$

In the equation $\tilde{y}(t+d)$ implies a prediction of the output after the time-delay, which takes into consideration the error between the model and the actual plant. Therefore, the integration is also made by ignoring the time-delay.

The final configuration of the control is shown in Fig. 7.6.9. Fig. 7.6.10 shows the response of this control. We can see the steady state error and the noisy input signal are well reduced, compared with the case in Fig. 7.6.8. In this simulation we take 0.2, 0.5, and 0.02 as the gain of feed-back linearization k, the factor of filter k_f and the factor of integrator k_i respectively.

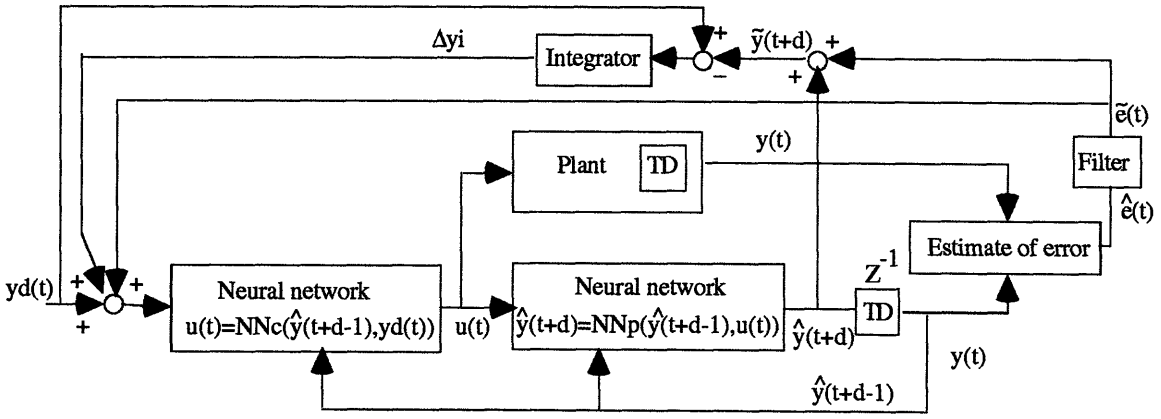


Fig. 7.6.9 Block diagram of control by modified Smith's principle

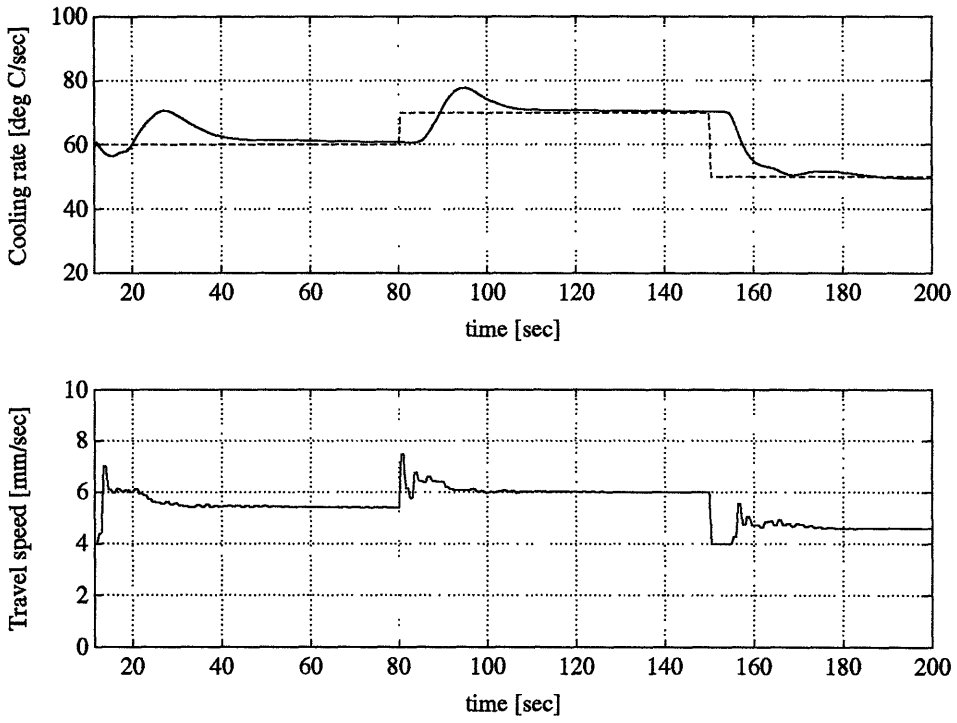


Fig. 7.6 10 Control by modified Smith's principle for disturbance 10[°c/sec]

7.6.5 Comparison with other methods

Figure 7.6.11 shows the final result of the control using neural networks. In this figure the result by PI control is also shown. We can see the control using neural networks is quite satisfactory. The settling time is reduced to about one third of PI control case. This result is better than the adaptive control result shown in Fig. 6.4.11.

The oscillatory control action is observed. This is the inherent nature of this control, which is caused by forcibly linearizing the nonlinear plant. Even if we make the response slower by decreasing the linearization factor k, we can not avoid the oscillation as shown in Fig. 7.6.5.

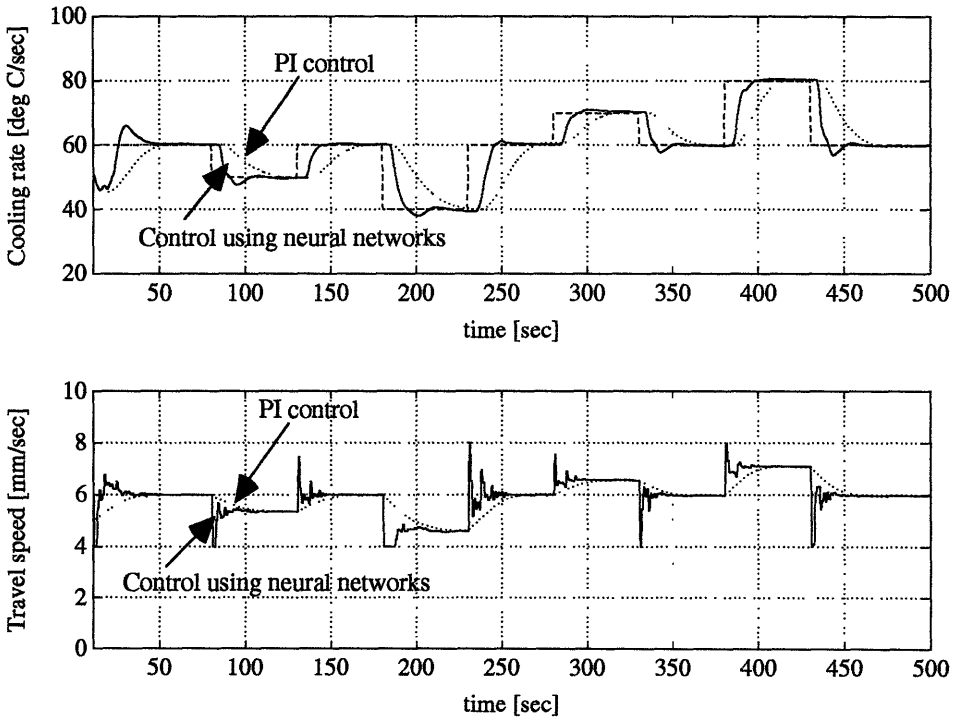


Fig. 7.6.11 Comparison between control using neural networks and PI control

7.7 Concluding remarks

Neural networks are quite attractive from the view-point that they can approximate any nonlinear system. However, they have the following limitations:

1. The identification by back-propagation is quite slow.
2. We can not say anything about the extrapolation.
3. To get a satisfactory approximation, the sample data should be well distributed in the region of interest because of **2**.
4. We need a large number of experiments for the identification due to **3**.

Many researchers have proposed many structures of controls using neural networks for a decade. In particular the nonlinear adaptive scheme has raised much interest, because we can possibly avoid the laborious experiments. However, the slow convergence is fatal for this adaptive scheme. As far as we study the performance of neural networks, we think we can not avoid the slow convergence, because the slow convergence seems to be the inherent nature of neural networks. From this point of view, it would be impossible to apply the adaptive scheme using neural networks.

The limitations **3** and **4** also give a big constraint on the application of neural networks. The fundamental problem lies in the fact we do not have any established methods for the experiments. We do not have a theory like a persistent excitation for linear systems. Furthermore the increase of the number of the experiments increases the number of sample data. That results in the increase of the number of computations necessary to get a satisfactory identification. Therefore, the identification problem would be further complicated if we consider a high order nonlinear system.

To avoid the problems above we assumed a first order nonlinear system for the cooling rate. Fortunately, the assumption is not so bad, considering the nature of the cooling rate. Although the design requires laborious computations by the parallel identification based on the conjugate gradient minimization method and the back-propagation with a distal system, it was rather simple because of the simple form of the model. We can also evaluate the identified system by visualizing the mapping by neural networks

To deal with a time-delay, we used the control based on Smith's method. The strong point of this configuration is that the neural networks are used as a nonlinear feed-forward controller to linearize the plant dynamics by canceling its nonlinearity. This feed-forward configuration is quite attractive, because we do not have to be concerned with any uncertainties that take place in the plant. This control is regarded as a control to

enhance the capacity of the feed-back control. Although there are some critics against Smith's principle, we showed this control scheme works quite well for the cooling rate.

The methods we have used may be applied to other nonlinear systems. The idea of our methods is as follows.

1. Use neural networks as a nonlinear first order model. If the system is of higher order, just pay attention to the slowest mode.
2. Identify the neural network by the parallel identification method.
3. Design the controller by the back-propagation with a distal system.
4. Embed the controller as a feed-forward controller which cancels the nonlinearity of the system.
5. Design the feed-back loop.

What we want to stress in this configuration is the linearization of the plant dynamics by the feed-forward loop where neural networks are embedded. The success of this method may depend on the accuracy of the approximation by the nonlinear model and the amount of the uncertainties given to the system. It is noted again we are not worried about the uncertainties when we design the feed-forward loop using neural networks. Obviously, it is difficult to design the feed-back loop, because we can not exactly describe how the resulting system behaves. However, suppose we simply use a PI controller for the feed-back loop. Then, the PI controller must show better performance than the conventional PI control, because the system is almost linearized by a nonlinear feed-forward control.

Chapter 8

Comparison between adaptive control and control using neural networks

We have so far discussed two control methods, adaptive control and control using neural networks, focusing on the design of the controllers. This chapter is more concerned with the performance of both controls. We also discuss the difference of implementation and tuning.

Command following

As discussed in Section 6.4.2 for adaptive control and in Section 7.6.5 for control using neural networks, both control schemes show better performance than PI controller for the command following. However, they yield the oscillatory control action while the control action of PI control is quite smooth. Therefore, if the oscillation in the control signal should be avoided, we have to degrade the performance or we cannot choose but PI control. Thus, both controls have a trade-off between the performance and the control action.

Disturbance rejection

Figure 8.1.1 shows the performance of the three control methods, adaptive control, control using neural networks and PI control, when the artificial disturbance of sinusoidal signal with the cycle time of 40 sec and the magnitude of 10 °C/sec is added to the cooling rate. In the adaptive control the control parameters λ and γ take 0.03 and 0.8 respectively, and the control using neural networks uses 0.2, 0.5, and 0.02 as the control parameters k, k_f and k_i respectively.

The adaptive control and the control using neural networks show better performance for disturbance rejection than PI control. Although the adaptive control shows a little better performance than the control using neural networks, both controls make a similar control action.

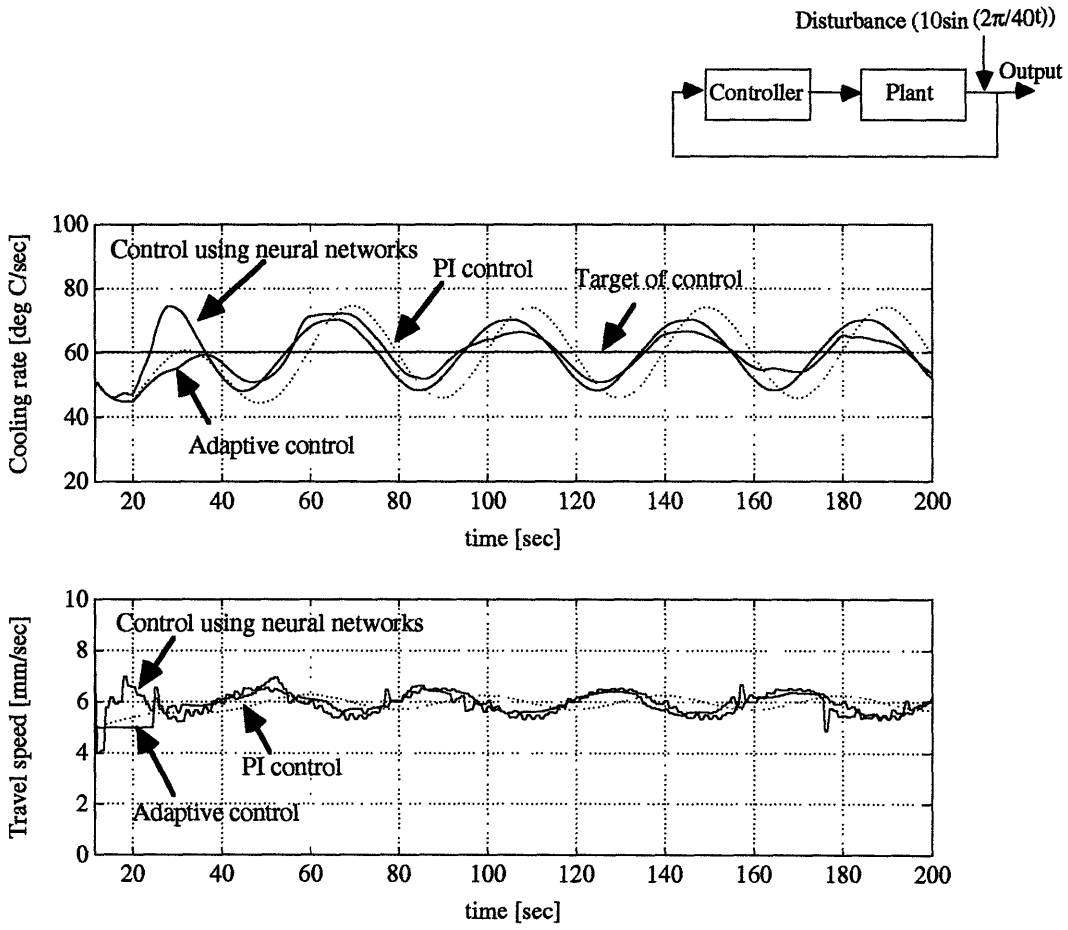
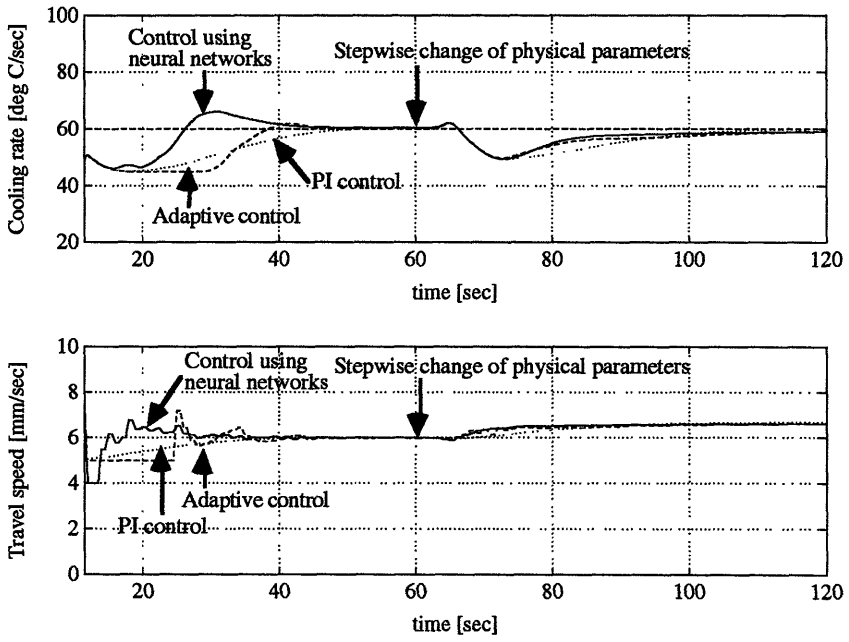


Fig. 8.1.1 Disturbance rejection by three control methods

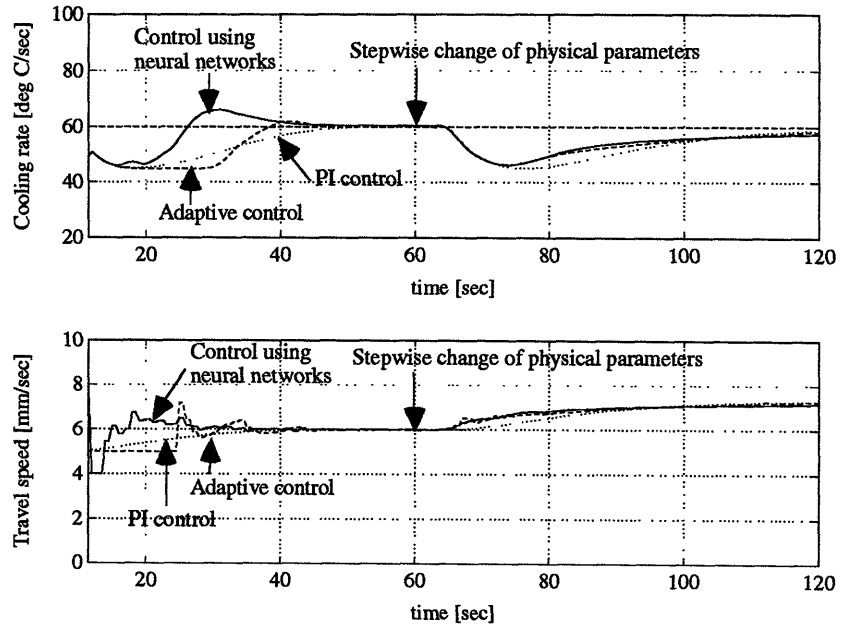
Insensitivity to perturbation of plant

Figures 8.1.2 through 8.1.5 show how the three controllers perform when the plant is perturbed. In the figures the physical parameters, the thermal diffusivity and the thickness, are stepwisely changed by ± 10 % at the time of 60 sec. The control using neural networks and the adaptive control show better performance for all the cases except the case of the thickness increased by 10 % and the thermal diffusivity decreased by 10 %, where the adaptive control brings about oscillatory response. The overall performance of the control using neural networks better than the adaptive control. When the effect of the perturbation increases the cooling rate, the adaptive control and the control using neural networks perform much better than the PI control. However, when the effect of the perturbation decreases the cooling rate, there is no significant difference among the three

controls, although the control using neural networks shows slightly better performance. In all cases the oscillatory control action is not observed.

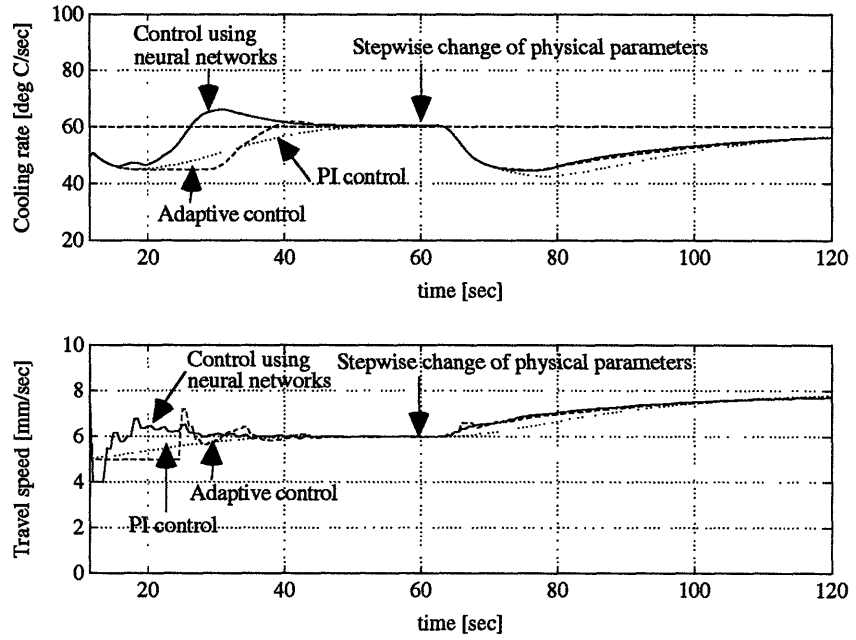


(Decrease of thickness by 10%, and decrease of thermal diffusivity by 10%)

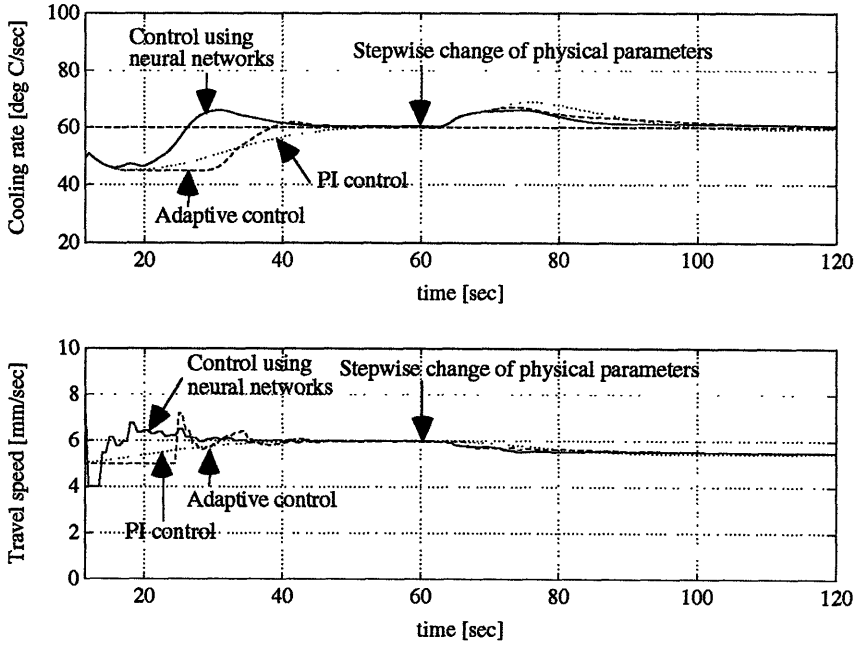


(Decrease of thickness by 10%)

Fig. 8.1.2 Performance for plant perturbation (a)

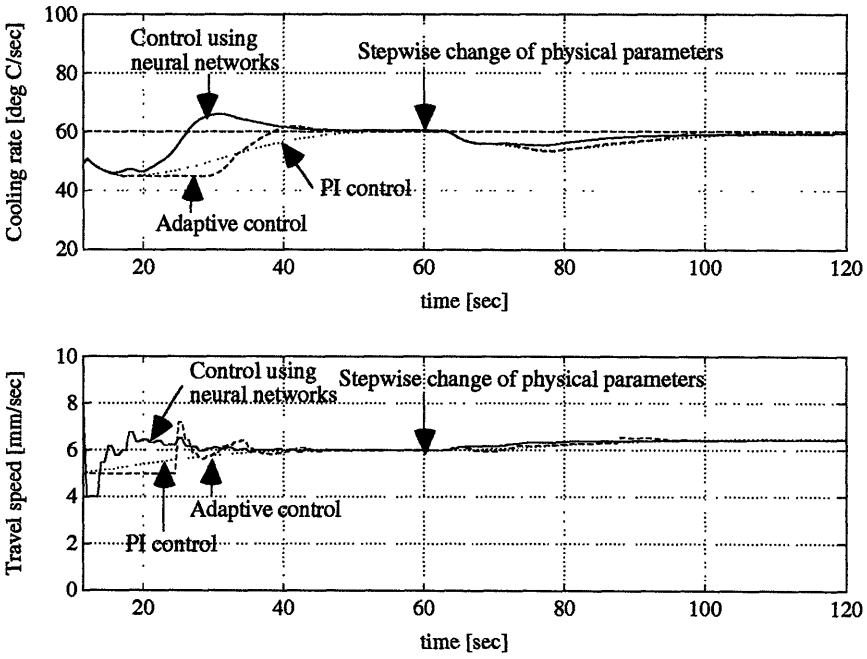


(Decrease of thickness by 10%, and increase of thermal diffusivity by 10%)

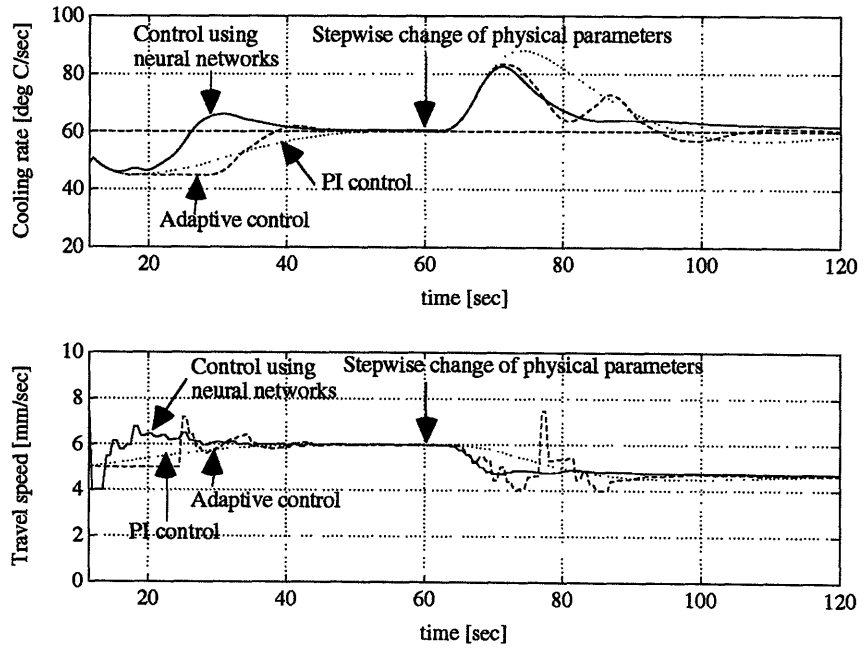


(Decrease of thermal diffusivity by 10%)

Fig. 8.1.3 Performance for plant perturbation (b)

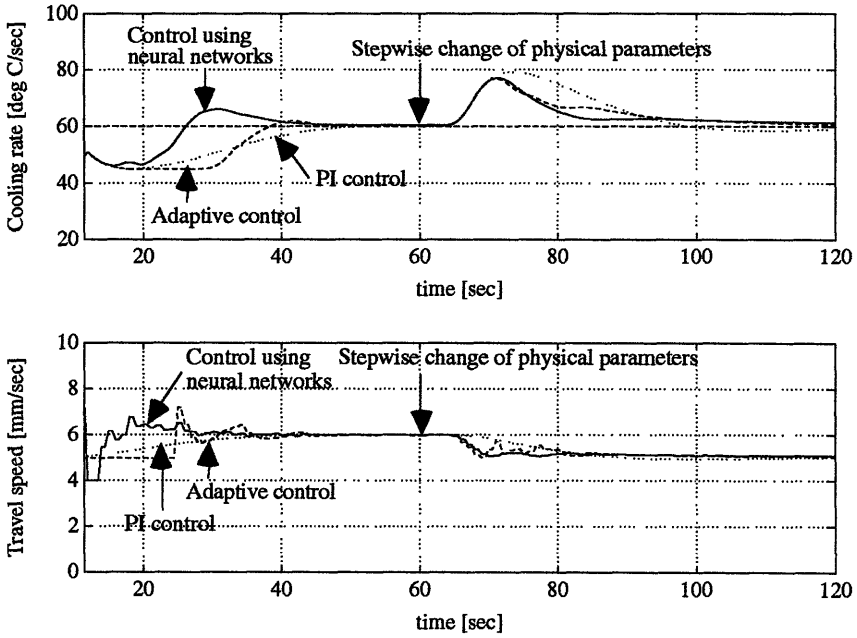


(Increase of thermal diffusivity by 10%)

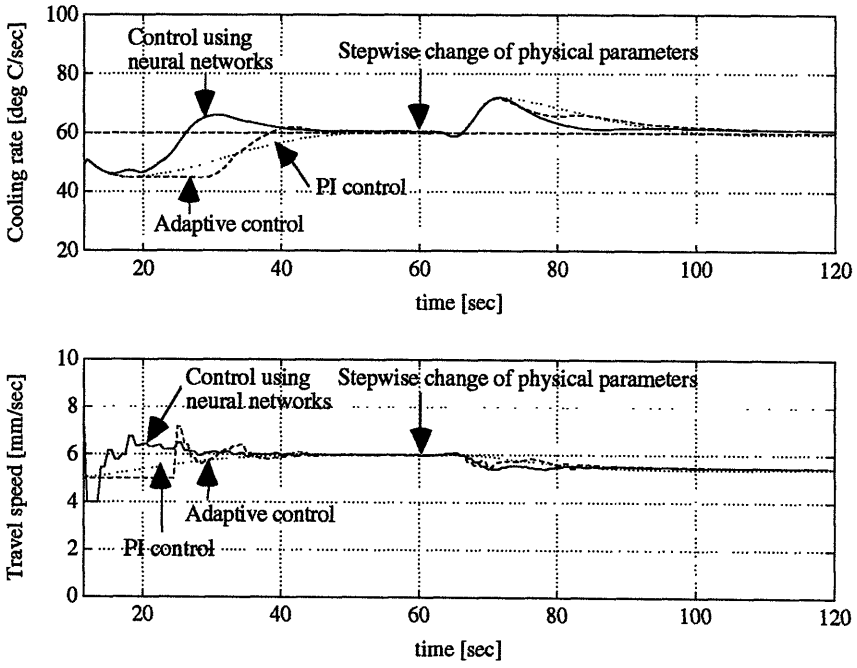


(Increase of thickness by 10%, and decrease of thermal diffusivity by 10%)

Fig. 8.1.4 Performance for plant perturbation (c)



(Increase of thickness by 10%)



(Increase of thickness by 10%, and increase of thermal diffusivity by 10%)

Fig. 8.1.5 Performance for plant perturbation (d)

Implementation

The implementation of adaptive control is considerably simpler than that of control using neural networks. Both controller designs require experiments before the design. In the case of the adaptive control, they are conducted simply to determine the model structure. However, in the case of the control using neural networks, we have to identify the nonlinear model by the experiments. As mentioned in Chapter 7, the identification requires a rather large number of experiments even if the system is of first order.

Once we determine the model structure, we can easily design the algorithm of the adaptive control in the manner discussed in Chapter 6. In the control using neural networks, we have to construct a nonlinear model by the obtained experimental data. That requires a large number of computations, which sometimes gives us a pain in the neck. Furthermore we also have to construct the nonlinear controller by a neural network. Again we need a large number of computations.

However, in the case of the control using neural networks, we can estimate how the nonlinear controller behaves by combining the two neural networks, once we get the neural network representing the plant and the one representing the nonlinear controller. We can also ensure the stability by the simulation. In the case of the adaptive control, we can not estimate anything until we actually apply the control to the process.

Tuning

The tuning of the control using neural networks is simpler than the adaptive control. As mentioned above we can estimate behavior of the control using neural networks before the actual application. In other words, we can understand what may happen in the controller in the actual application. Therefore, when we tune the parameters, we can easily find what we should do. However, in the case of the adaptive control, it is quite hard to understand what takes place inside the controller. The control may be unstable because of the bad selection of the model structure and the control law.

Chapter 9

Conclusions

9.1 Future work

This thesis mainly focuses on the design issues of the nonlinear control for the cooling rate. The evaluation of the control is made only by simulation. Therefore, we have not addressed the performance for noise, which is essential to the process control. The final evaluation of the performance of control should be made by the actual application of the designed control.

Final objective of this research is to establishment the control for the cooling rate and the mass transfer rate. The control of the mass transfer rate is quite simple. It simply changes the wire feed rate in accordance with the change of the travel speed, which results from the control of the cooling rate. This configuration is based on the fact the cooling rate is not affected by the wire feed rate. Therefore, we have to ensure whether this configuration truly works by the actual application.

9.2 Conclusions

This thesis contributes to the control of the cooling rate in gas metal arc welding. It copes with the measurement system, the analysis of the steady-state and transient behavior of the cooling rate, the derivation a model which describes the cooling rate dynamics, and the selection and design of the nonlinear control system.

The measurement system using an infrared pyrometer which scans the weldment using an oscillating mirror is established to measure the temperature distribution on the weldment during GMAW. The measurement reveals that wire feed rate has little effect on the steady-state cooling rate. By contrast, the travel speed has a strong effect on cooling rate. This observation implies that cooling rate and mass transfer rate can be independently controlled in-process.

Extensive derivation of dynamic models for the temperature during welding is made by the Green's function method. A comparison between analytical models and the actual temperatures shows a discrepancy. The experiments indicate the steady-state temperature distribution can be described as an exponential function of the distance from

the arc for the temperature range of interest (600 °C to 1000 °C). The analytical models do not succeed in explaining such behavior. On the other hand, there is good correspondence between the analytical model and actual measurement for the dynamic behavior of the cooling rate.

The experimental results and the analytical models suggest the following strategies for the control of the cooling rate.

1. The cooling rate should be controlled by travel speed.
2. We should design the control to accommodate a nonlinearity and a big time-delay.

The PI controller works for the control of the cooling rate. However, the performance is not satisfactory.

Nonlinear controls are extensively reviewed. The review proposes two types of controls, linear adaptive control and nonlinear control using neural networks. The adaptive control adjusts the parameters of its linear model to the perturbation caused by the nonlinearity. Control using neural networks regulates the cooling rate by directly employing a nonlinear model. Thus, both schemes can accommodate the nonlinear nature of the process.

A one-step-ahead adaptive control method is used as the adaptive controller. Practical considerations, such as a forgetting factor, an integrator and a constrained parameter estimation are discussed. A recursive least-square algorithm is used as the on-line estimator. The estimation by this algorithm is fast enough to accommodate the nonlinearity of the process. Two types of model structures are proposed to cope with the varying time-delay. Type 2 method shows a satisfactory result for the varying time-delay. This method employs some model structures in accordance with a variation of the time-delay, and simultaneously estimates all the parameters of these structures on-line. The model that gives the best prediction is selected as the plant model used for the control law.

For the controller using neural networks, first the characteristics of neural networks are investigated. Some limitations that come from the complexity of neural computation and nonlinear identification are found. The limitations recommend a control based on a feed-back scheme with fixed parameters. It seems that a control based on an adaptive scheme is not applicable due to the fundamental drawback, or the slow estimation of the parameters in neural networks. The investigation also suggests Smith's principle to deal with the time-delay. Design methods, such as the parallel identification based on a conjugate gradient algorithm and the controller design based on a back-propagation method with a distal system, are proposed. The Smith's principle brings

about the configuration composed of two loops, inner loop and outer loop. Two neural networks, which represent the plant model and the nonlinear controller, are employed in the inner loop, playing a role of canceling a nonlinearity of the process as a feed-forward controller. All uncertain factors such as disturbance and perturbation of the plant are accommodated by the outer loop. Some considerations such as an integrator and a filtering of an error estimate are proposed based on the performance of the nonlinear controller.

Both control schemes bring about quite satisfactory results, showing better performance for command following, disturbance rejection, and insensitivity to plant perturbation than PI control. However, they often yield oscillatory control action. Therefore, there is a trade-off between the performance and the control action. We need further investigation of the performance by the actual application to the process.

Appendix A

Derivation of temperature distribution model during welding

A.1 Partial differential equation for heat transfer during welding

Conduction heat transfer is described as the partial differential equation[1] :

$$c\rho\frac{\partial T(x,y,z,t)}{\partial t} = q(x,y,z,t) + \lambda(\frac{\partial}{\partial x^2} + \frac{\partial}{\partial y^2} + \frac{\partial}{\partial z^2})T(x,y,z,t) \qquad \text{(a.1)}$$

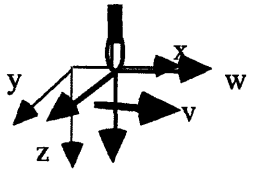
If the heat source is moving at a velocity $v(t)$, (a.1) yields the temperature in the moving coordinate is given by:

$$L[T] \equiv k(\frac{\partial}{\partial w^2} + \frac{\partial}{\partial y^2} + \frac{\partial}{\partial z^2})T + v(t)\frac{\partial T}{\partial w} - \frac{\partial T}{\partial t} = -\frac{q(w,y,z,t)}{c\rho} \qquad \text{(a.2)}$$

where $k = \dfrac{\lambda}{c\rho}$

since

$$x = w + \int_0^t v(\tau)d\tau$$



$$\frac{\partial^2 T(w,y,z,t)}{\partial x^2} = \frac{\partial^2 T}{\partial w^2}$$

$$\frac{\partial T(w(t),y,z,t)}{\partial t} = \frac{\partial T(w,y,z,t)}{\partial t} + \frac{\partial w}{\partial t}\frac{\partial T(w,y,z,t)}{\partial w} = \frac{\partial T(w,y,z,t)}{\partial t} - v(t)\frac{\partial T(w,y,z,t)}{\partial w}$$

The equation (a.2) is fundamental for describing the temperature distribution during welding.

A.2 Green's function

Consider a partial differential equation given by:

$$L[u] = f(w,y,z,t) \qquad \text{(a.3)}$$

where L is an operator which is described by

$$L = L(\frac{\partial}{\partial w}, \frac{\partial}{\partial w^2}, \frac{\partial}{\partial y}, \frac{\partial}{\partial y^2}, \frac{\partial}{\partial z}, \frac{\partial}{\partial z^2}, \frac{\partial}{\partial t} \cdots)$$

---

[1] In this equation, thermal diffusivity is assumed to be constant. In actual process, it is dependent on the temperature. If we do not ignore the dependence , the equation should be described as:

$$c\rho\frac{\partial T(x,y,z,t)}{\partial t} = Q(x,y,z,t) + [\frac{\partial}{\partial x}(\lambda\frac{\partial T}{\partial x}) + \frac{\partial}{\partial y}(\lambda\frac{\partial T}{\partial y}) + \frac{\partial}{\partial z}(\lambda\frac{\partial T}{\partial z})]$$

Green's function $G(w,y,z,t,\xi,\psi,\zeta,\tau)$ is the solution for

$$L[G]=\delta(w-\xi)\cdot\delta(y-\psi)\cdot\delta(z-\zeta)\cdot\delta(t-\tau) \tag{a.4}$$

under given boundary conditions. If we can get the Green's function, the solution for the equation (a.3) is given by:

$$u(w,y,z,t)=\iiint\int G(w,y,z,t,\xi,\psi,\zeta,\tau)f(\xi,\psi,\zeta,\tau)d\xi d\psi d\zeta d\tau \tag{a.5}$$

based on the principle of superposition. It is noted that Green's function is not unique, depending on the boundary conditions. There are some methods to obtain Green's function as shown below if the coefficients in L are constant.

Green's function for infinite space

Using Fourier transformation we can get

$$\begin{aligned} h(w,y,z,t)=&(\frac{1}{2\pi})^4\iiint\int_{-\infty}^{\infty}d\omega_w d\omega_y d\omega_z d\omega_t\iiint\int_{-\infty}^{\infty}dw'\,dy'\,dz'\,dt'\,h(w',y',z',t')\\ &\cdot\exp(i\omega_w(w-w'))\exp(i\omega_y(y-y'))\exp(i\omega_z(z-z'))\exp(i\omega_t(t-t')) \end{aligned} \tag{a.6}$$

where $h(w,y,z,t)$ is an integrable function. Therefore,

$$\begin{aligned} \delta(w-\xi)\cdot\delta(y-\psi)\cdot\delta(z-\zeta)\cdot\delta(t-\tau)=&(\frac{1}{2\pi})^4\iiint\int_{-\infty}^{\infty}d\omega_w d\omega_y d\omega_z d\omega_t\\ &\cdot\exp(i\omega_w(w-\xi))\exp(i\omega_y(y-\psi))\exp(i\omega_z(z-\zeta))\exp(i\omega_t(t-\tau)) \end{aligned} \tag{a.7}$$

The comparison of (a.4) and (a.7) gives us the following Green's function:

$$\begin{aligned} G(w,y,z,t,x,y,z,t)=&(\frac{1}{2\pi})^4\iiint\int_{-\infty}^{\infty}d\omega_w d\omega_y d\omega_z d\omega_t g(\omega_w,\omega_y,\omega_z,\omega_t)\\ &\cdot\exp(i\omega_w(w-\xi))\exp(i\omega_y(y-\psi))\exp(i\omega_z(z-\zeta))\exp(i\omega_t(t-\tau)) \end{aligned} \tag{a.8}$$

$$g(\omega_w,\omega_y,\omega_z,\omega_t)=\frac{\exp(i\omega_w(w-\xi))\exp(i\omega_y(y-\psi))\exp(i\omega_z(z-\zeta))\exp(i\omega_t(t-\tau))}{L(\exp(i\omega_w(w-\xi))\exp(i\omega_y(y-\psi))\exp(i\omega_z(z-\zeta))\exp(i\omega_t(t-\tau)))}$$

It is noted that this method should be applied to the infinite space, because (a.6) involves the infinite integrals.

Green's function for finite space

Consider the space is limited to a boundary in z direction. If we can describe the solution as the series of a complete set of orthonormal functions like Fourier series:

$$u(w,y,z,t)=\sum_n u_n(w,y,t)\phi_n(z)$$

$\phi_n(z)$: Orthonormal functions that satisfy the boundary conditions

then

$$\delta(z-\zeta)=\sum_n \phi_n(z)\phi_n(\zeta) \tag{a.9}$$

The substitution of (a.9) into (a.7) and the comparison of (a.4) and (a.7) gives us the Green's function:

$$\begin{aligned} G(w,y,z,t,\xi,\psi,\zeta,\tau)&=(\frac{1}{2\pi})^3\sum_n\iiint_{-\infty}^{\infty} d\omega_w d\omega_y d\omega_t g_n(\zeta,\omega_w,\omega_y,\omega_t) \\ &\cdot\exp(i\omega_w(w-\xi))\exp(i\omega_y(y-\psi))\exp(i\omega_t(t-\tau))\phi_n(z) \\ g_n(\zeta,\omega_w,\omega_y,\omega_t)&= \\ &\frac{\exp(i\omega_w(w-\xi))\exp(i\omega_y(y-\psi))\exp(i\omega_t(t-\tau))\phi_n(z)}{L(\exp(i\omega_w(w-\xi))\exp(i\omega_y(y-\psi))\exp(i\omega_t(t-\tau))\phi_n(z))}\phi_n(\zeta) \end{aligned} \tag{a.10}$$

We can apply the similar method to the case the space has boundary limits in other axes.

A.3 The solution for infinite space and point heat source

Green's function

We assume that the transfer speed v is kept constant. From (a.2)

$$L=k(\frac{\partial^2}{\partial w^2}+\frac{\partial^2}{\partial y^2}+\frac{\partial^2}{\partial z^2})+v\frac{\partial}{\partial w}-\frac{\partial}{\partial t} \tag{a.11}$$

Therefore, $g(\omega_w,\omega_y,\omega_z,\omega_t)$ in (a.8) is given by

$$\begin{aligned} &g(\omega_w,\omega_y,\omega_z,\omega_t) \\ &=\frac{1}{-k(\omega_w{}^2+\omega_y{}^2+\omega_z{}^2)+iv\omega_w-i\omega_t} \end{aligned} \tag{a.12}$$

The equation (a.8), therefore, yields the Green's function:

$$\begin{aligned} G(w,y,z,t,\xi,\psi,\zeta,\tau)&=(\frac{1}{2\pi})^4\iiint_{-\infty}^{\infty} d\omega_w d\omega_y d\omega_z d\omega_t \frac{-1}{k(\omega_w{}^2+\omega_y{}^2+\omega_z{}^2)-iv\omega_w+i\omega_t} \\ &\cdot\exp(i\omega_w(w-\xi))\exp(i\omega_y(y-\psi))\exp(i\omega_z(z-\zeta))\exp(i\omega_t(t-\tau)) \end{aligned} \tag{a.13}$$

Consider the following complex integral:

$$\int_{-\infty}^{\infty} d\omega_t \frac{1}{A+i\omega_t}\exp(i\omega_t(t-\tau)) = \begin{cases} 2\pi\exp(-A)(t-\tau) & (t \geq \tau) \\ 0 & (t < \tau) \end{cases} \tag{a.14}$$

Then, the equation (a.13) yields

$$\begin{aligned} &G(w,y,z,t,\xi,\psi,\zeta,\tau) \\ &= -(\frac{1}{2\pi})^3 \iiint \int_{-\infty}^{\infty} d\omega_w d\omega_y d\omega_z \exp((-k(\omega_w{}^2+\omega_y{}^2+\omega_z{}^2)+iv\omega_w)(t-\tau) \\ &\cdot\exp(i\omega_w(w-\xi))\exp(i\omega_y(y-\psi))\exp(i\omega_z(z-\zeta)) \qquad (t>\tau) \end{aligned} \tag{a.15}$$

Since $\int_{-\infty}^{\infty}\exp(-ax^2) = \sqrt{\frac{\pi}{a}}$

the equation (a.15) gives us

$$\begin{aligned} &G(w,y,z,t,\xi,\psi,\zeta,\tau) \\ &= \begin{cases} -(\frac{1}{2\sqrt{\pi}\sqrt{k}})^3(\frac{1}{\sqrt{t-\tau}})^3\exp(-\frac{(w-\xi+v(t-\tau))^2+(y-\psi)^2+(z-\zeta)^2}{4k(t-\tau)}) & (t>\tau) \\ 0 \ (t<\tau) \end{cases} \end{aligned} \tag{a.16}$$

Solution for semi-infinite space

Here we are concerned with the conditions: **(1)** Point heat source, **(2)** Semi-infinite and **(3)** No heat flux from the surface, which let the Green's function satisfy the conditions:

$$\begin{aligned} &L(G(w,y,z,t,\xi,\psi,\zeta,\tau)) = \delta(w-\xi)\delta(y-\psi)\delta(z-\zeta)\delta(t-\tau) \qquad (z\leq 0, \zeta \leq 0) \\ &\left|\frac{\partial G(w,y,z,t,\xi,\psi,\zeta,\tau)}{\partial z}\right|_{z=0} = 0 \end{aligned} \tag{a.17}$$

For this problem we can use the following Green's function by the mirror method.

$$G(w,y,z,t,\xi,\psi,\zeta,\tau) = G_0(w,y,z,t,\xi,\psi,\zeta,\tau) + G_0(w,y,z,t,\xi,\psi,-\zeta,\tau) \tag{a.18}$$

where $G_0(w,y,z,t,\xi,\psi,\zeta,\tau)$ is the Green's function which is described in the equation (a.16). We can easily see that the Green's function written as (a.18) satisfies the conditions (a.17).

Since point heat source is given by:

$$f(w,y,z,t) = -\frac{q(w,y,z,t)}{c\rho} = -\frac{q(t)}{c\rho}\delta(w)\delta(y)\delta(z) \tag{a.19}$$

the substitution of (a,16), (a.18) and (a.19) into (a.5) yields the temperature distribution:

$$\begin{aligned} &T(w,y,z,t) = \\ &\int_0^t d\tau[2\cdot(\frac{1}{2\sqrt{\pi}\sqrt{k}}\frac{1}{\sqrt{t-\tau}})^3\frac{q(\tau)}{c\rho}\exp(-\frac{(w+v(t-\tau))^2+y^2+z^2}{4k(t-\tau)}) \end{aligned} \tag{a.20}$$

For the steady-state temperature distribution, where the heat input are kept constant at Q making time t infinite, we obtain

$$T(w,y,z,\infty)=\int_0^\infty ds[\frac{1}{4}(\frac{1}{\sqrt{\pi}\sqrt{k}})^3\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\exp(-\frac{s^2}{4k}(w^2+y^2+z^2)-\frac{1}{s^2}\frac{v^2}{4k})$$
$$=\frac{Q}{2\pi\lambda}\frac{1}{r}\exp(-\frac{1}{2k}vr)\exp(-\frac{1}{2k}vw) \qquad \text{(a.21)}$$
$$s=\frac{1}{\sqrt{t-\tau}}\quad k=\lambda/c\rho\quad r=\sqrt{w^2+y^2+z^2}$$

since $\int_0^\infty \exp(-a^2s^2-\frac{b^2}{s^2})=\frac{\sqrt{\pi}}{2a}\exp(-2ab)$

This is Rosenthal's model.

A.4 The solution for finite thickness with heat flux from surface

Green's function

Here we are concerned with the conditions in which the workpiece has a finite thickness and heat flux can not be ignored. Then, the boundary conditions can be described as:

$$\left[\frac{\partial T(w,y,z,t)}{\partial z}\right]_{z=0}=h\cdot(T(w,y,0,t)-T_0)$$
$$\left[\frac{\partial T(w,y,z,t)}{\partial z}\right]_{z=a}=-h\cdot(T(w,y,a,t)-T_0) \qquad \text{(a.22)}$$

We need look for the orthogonal functions which satisfy the boundary conditions. In a similar manner as Fourier series, we assume the function can be written as:

$$\phi_n(z)=\sin(\alpha z+\beta) \qquad \text{(a.23)}$$

From (a.22)

$$\alpha\cos(\alpha a+\beta)+h\sin(\alpha a+\beta)=0$$
$$\alpha\cos(\beta)-h\sin(\beta)=0 \qquad \text{(a.24)}$$

Therefore,

$$\cos(\alpha a+\beta)\sin(\beta)+\sin(\alpha a+\beta)\cos(\beta)=\sin(\alpha a+2\beta)=0 \qquad \text{(a.25)}$$

The equations (a.23) and (a.25) bring about a set of functions:

$$\phi_n(z)=\sin(\alpha_n z+\beta_n)$$
$$\begin{cases}\alpha_n a+2\beta_n=n\pi\\ \tan(\beta_n)=\dfrac{\alpha_n}{h}=\dfrac{n\pi-2\beta_n}{ah}\end{cases} \qquad \text{(a.26)}$$

We can see these functions are orthogonal in the similar manner as Fourier series, since

$$\int_0^a \phi_n(z)\phi_m(z)dz = \begin{cases} \frac{1}{2}a + \frac{\cos^2(\beta_n)}{h} & (n=m) \\ 0 & (n \neq m) \end{cases} \tag{a.27}$$

By normalizing the functions, we obtain the following complete set of orthonormal functions for the given boundary conditions.

$$\phi_n(z) = \sqrt{\frac{2}{a + \frac{2\cos^2(\beta_n)}{h}}} \sin(\alpha_n z + \beta_n)$$
$$\begin{cases} \alpha_n a + 2\beta_n = n\pi \\ \tan(\beta_n) = \frac{\alpha_n}{h} = \frac{n\pi - 2\beta_n}{ah} \end{cases} \tag{a.28}$$

The insertion of (a.28) into (a.10) yields

$$G(w,y,z,t,x,y,z,t) = (\frac{1}{2\pi})^3 \sum_n \iiint_{-\infty}^{\infty} d\omega_w d\omega_y d\omega_t g_n(\zeta,\omega_x,\omega_y,\omega_z,\omega_t)$$
$$\cdot \exp(i\omega_w(x-\xi))\exp(i\omega_y(y-\psi))\exp(i\omega_t(t-\tau))\phi_n(z) \tag{a.29}$$
$$g_n(\zeta,\omega_w,\omega_y,\omega_z,\omega_t) = \frac{\phi_n(\zeta)}{-k(\omega_w{}^2 + \omega_y{}^2 + \alpha_n{}^2) + iv\omega_w - i\omega_t}$$

By the similar integration shown in (a.14), the equation (a.29) can be rewritten as:

$$G(w,y,z,t,\xi,\psi,\zeta,\tau)$$
$$= \begin{cases} -\frac{1}{4\pi k}\sum_n^{\infty} \frac{1}{t-\tau} \exp(-\frac{(w-\xi+v(t-\tau))^2 + (y-\psi)^2}{4k(t-\tau)}) \exp(-k\alpha_n{}^2(t-\tau))\phi_n(z)\phi_n(\zeta) & (t>\tau) \\ 0 & (t<\tau) \end{cases} \tag{a.30}$$

The solution for point heat source

Using the Green's function shown in (a.30), the equations (a.5) and (a.19) yield

$$T(w,y,z,t) =$$
$$\frac{1}{4\pi k}\sum_{n=1}^{\infty} \int_0^t d\tau \frac{1}{t-\tau} \frac{q(\tau)}{c\rho} \exp(-\frac{(w+v(t-\tau))^2 + y^2}{4k(t-\tau)} - k\alpha_n{}^2(t-\tau))\phi_n(z)\phi_n(0) \tag{a.31}$$

If the heat input is kept constant at Q, the equation (31) can be described as:

$$T(w,y,z,t) =$$
$$\frac{1}{4\pi k}\frac{Q}{c\rho} \exp(-\frac{wv}{2k}) \sum_{n=1}^{\infty} \int_0^t d\tau \frac{1}{t-\tau} \exp(-\frac{w^2+y^2}{4k(t-\tau)} - (\frac{v^2}{4k} + k\alpha_n{}^2)(t-\tau))\phi_n(z)\phi_n(0) \tag{a.32}$$

Therefore, the steady-state solution can be obtained by:

$$T(w,y,z,\infty)$$

$$= \frac{1}{2\pi k}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\sum_{n=1}^{\infty}\lim_{\nu\to\infty}\int_{\frac{1}{\sqrt{t}}}^{\infty}\frac{1}{s}\exp(-\frac{w^2+y^2}{4k}s^2-(\frac{v^2}{4k}+k\alpha_n{}^2)\frac{1}{s^2})ds\phi_n(z)\phi_n(0)$$

$$= \frac{1}{\pi k}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\sum_{n=1}^{\infty}K_0(\frac{\sqrt{v^2+4k\alpha_n{}^2}}{2k}\sqrt{w^2+y^2})\phi_n(z)\phi_n(0) \quad (s=\frac{1}{\sqrt{t-\tau}}) \tag{a.33}$$

K_0: Modified Bessel function of the second kind and zero order

since $\int_0^{\infty}\frac{1}{s}\exp(-a^2s^2-b^2\frac{1}{s^2})ds = 2K_0(2ab)$

The solution for Gaussian distributed heat input

The Gaussian heat source is described by

$$f(w,y,z,t) = -\frac{q(t)}{c\rho}\frac{1}{2\pi\sigma^2}\exp(-\frac{w^2+y^2}{2\sigma^2})\delta(z) \tag{a.34}$$

The insertion of (a.34) and (a.30) into (a.5) yields

$$T(w,y,z,t) =$$

$$\frac{1}{4\pi k}\frac{1}{2\pi\sigma^2}\sum_{n=1}^{\infty}\int_{-\infty}^{\infty}\int d\xi d\psi\int_0^t d\tau\frac{1}{t-\tau}\frac{q(\tau)}{c\rho}\exp[-\frac{(w-\xi+v(t-\tau))^2+(y-\psi)^2}{4k(t-\tau)}-\frac{\xi^2+\psi^2}{2\sigma^2}-k\alpha_n{}^2(t-\tau)]$$

$$\cdot\phi_n(z)\phi_n(0) \tag{a.35}$$

Since $\int_{-\infty}^{\infty}\exp(-ax^2+bx+c) = \frac{\sqrt{\pi}}{\sqrt{a}}\exp(\frac{b^2}{4a}+c)$

$$T(w,y,z,t) =$$

$$\frac{1}{\pi}\sum_{n=1}^{\infty}\int_0^t\frac{1}{4k(t-\tau)+2\sigma^2}\frac{q(\tau)}{c\rho}\exp\left[-\frac{(w+v(t-\tau))^2+y^2}{4k(t-\tau)+2\sigma^2}-k\alpha_n{}^2(t-\tau)\right]d\tau\Phi_n(z)\Phi_n(0) \tag{a.36}$$

For the steady-state solution, we can not make the model in (a.35) simpler using a special function. We don't have such expressions as Bessel function to simplify the model

A.5 The solution for finite thickness, finite width and heat flux from the surface

We consider the temperature distribution under the conditions:

The workpiece has finite thickness and width, and the heat flux from the surface is governed by:

$$\left[\frac{\partial T(w,y,z,t)}{\partial z}\right]_{z=0} = h\cdot(T(w,y,0,t)-T_0)$$
$$\left[\frac{\partial T(w,y,z,t)}{\partial z}\right]_{z=a} = -h\cdot(T(w,y,a,t)-T_0)$$
$$\left[\frac{\partial T(w,y,z,t)}{\partial z}\right]_{y=\frac{b}{2}} = 0 \qquad \text{(a.37)}$$
$$\left[\frac{\partial T(w,y,z,t)}{\partial z}\right]_{y=-\frac{b}{2}} = 0$$

In this case the temperature distribution is assumed to be symmetric with respect to the weldment in the width direction, or y axis.

In the similar manner to the finite thickness case, we can obtain the following orthogonal functions.

(Thickness direction)

$\phi_n(z)$, which is shown in (a.28)

(Width direction)

$$\varphi_m(y) = \begin{cases} \sqrt{\frac{2}{b}}\cos(\frac{m\pi}{b}y) \\ \sqrt{\frac{1}{b}} \end{cases} \qquad \text{(a.38)}$$

Using $\phi_n(z)$ and $\varphi_m(y)$, we can get the Green's function for the point heat source:

$$G(w,y,z,t,x,y,z,t) = (\frac{1}{2\pi})^2 \sum_{\substack{n \\ m}} \int\int_{-\infty}^{\infty} d\omega_x d\omega_t g_n(\psi,\zeta,\omega_w,\omega_t)$$
$$\cdot\exp(i\omega_w(w-\xi))\exp(i\omega_t(t-\tau))\phi_n(z)\varphi_m(y) \qquad \text{(a.39)}$$
$$g_n(\psi,\zeta,\omega_w,\omega_t) = \frac{\phi_n(\zeta)\varphi_m(\psi)}{-k(\omega_w{}^2+\alpha_n{}^2+(\frac{m\pi}{b})^2)+iv\omega_w-i\omega_t}$$

The integration in (a.39) yields the following result:

$$G(w,y,z,t,\xi,\psi,\zeta,\tau)$$
$$= \begin{cases} -\frac{1}{2\sqrt{\pi k}}\sum_{\substack{n \\ m}}^{\infty}\frac{1}{\sqrt{t-\tau}}\exp(-\frac{(w-\xi+v(t-\tau))^2}{4k(t-\tau)})\exp(-k(\alpha_n{}^2+(\frac{m\pi}{b})^2)(t-\tau)) \\ \qquad\qquad \cdot\phi_n(z)\phi_n(\zeta)\varphi_m(y)\varphi_m(\psi) \quad (t>\tau) \\ 0 \qquad\qquad (t<\tau) \end{cases} \qquad \text{(a.40)}$$

From (a.40), the temperature can be written as:

$$T(w,y,z,t)=$$

$$\frac{1}{4\sqrt{\pi k}}\sum_{\substack{n=1\\m=0}}^{\infty}\int_0^t d\tau\frac{1}{\sqrt{t-\tau}}\frac{q(\tau)}{c\rho}\exp(-\frac{(w+v(t-\tau))^2}{4k(t-\tau)}-k(\alpha_n{}^2+(\frac{m\pi}{b})^2)(t-\tau)) \qquad \text{(a.41)}$$

$$\cdot\phi_n(z)\phi_n(0)\varphi_m(y)\varphi_m(0)$$

If the heat input is kept constant, the equation (a.41) can be described as:

$$T(w,y,z,t)=$$

$$\frac{1}{4\sqrt{\pi k}}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\sum_{\substack{n=1\\m=0}}^{\infty}\int_0^t d\tau\frac{1}{\sqrt{t-\tau}}\exp(-\frac{w^2}{4k(t-\tau)}-(\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b})^2))(t-\tau))$$

$$\cdot\phi_n(z)\phi_n(0)\varphi_m(y)\varphi_m(0)$$

$$=\frac{1}{2\sqrt{\pi k}}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\sum_{\substack{n=1\\m=0}}^{\infty}\int_{\frac{1}{\sqrt{t}}}^{\infty}\frac{1}{s^2}\exp\left[-\frac{w^2}{4k}s^2-(\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b/2})^2)\frac{1}{s^2}\right]ds \qquad \text{(a.42)}$$

$$\cdot\Phi_n(z)\Phi_n(0)\Psi_m(y)\Psi_m(0) \quad (s=\frac{1}{\sqrt{t-\tau}})$$

Therefore, the steady-state solution can be described by:

$$T(w,y,z,\infty)=\frac{1}{2\sqrt{\pi k}}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\sum_{\substack{n=1\\m=0}}^{\infty}\int_0^{\infty}\frac{1}{s^2}\exp\left[-\frac{w^2}{4k}s^2-(\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b/2})^2)\frac{1}{s^2}\right]ds$$

$$\cdot\Phi_n(z)\Phi_n(0)\Psi_m(y)\Psi_m(0)$$

$$=\frac{1}{4\sqrt{k}}\frac{Q}{c\rho}\exp(-\frac{wv}{2k})\sum_{\substack{n=1\\m=0}}^{\infty}\frac{\exp\left[-\frac{w}{\sqrt{k}}\sqrt{\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b/2})^2)}\right]}{\sqrt{\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b/2})^2)}} \qquad \text{(a.43)}$$

$$\cdot\Phi_n(z)\Phi_n(0)\varphi_m(y)\varphi_m(0)$$

For the Gaussian heat source shown in (a.34), we can obtain the following result using the similar manner to get (a.35).

$$T(w,y,z,t)=$$

$$\frac{1}{4\left(\sqrt{\pi}\right)^3\sqrt{k}}\frac{1}{\sigma^2}\sum_{\substack{n=1\\m=0}}^{\infty}\int_{-\infty}^{\infty}\exp(-\frac{\theta^2}{2\sigma^2})\psi_m(\theta)d\theta$$

$$\cdot\int_{-\infty}^{\infty}d\zeta\exp(-\frac{(w-\zeta)v}{2k}-\frac{\zeta^2}{2\sigma^2})\int_0^t d\tau\frac{1}{\sqrt{t-\tau}}\frac{q(\tau)}{c\rho} \qquad \text{(a.44)}$$

$$\cdot\exp\left[-\frac{(w-\zeta)^2}{4k}\frac{1}{t-\tau}-(\frac{v^2}{4k}+k(\alpha_n{}^2+(\frac{m\pi}{b/2})^2)(t-\tau)\right]\Phi_n(z)\Phi_n(0)\Psi_m(y)$$

A.6 The solution for varying velocity

We have so far discussed the case in which the velocity is constant. Here we deal with the case of a varying velocity. We can see that all temperature models involve the term of $v(t-\tau)$. This term shows the distance between the centers of the heat sources $q(\tau)$ and $q(t)$. Therefore, we can obtain the modified model for the varying velocity by just replacing $v(t-\tau)$ by $\int_{\tau}^{t} v(\eta)d\eta$. For example, the equation (31) is modified to the model:

$$T(w,y,z,t) =$$

$$\frac{1}{4\pi k}\sum_{n=1}^{\infty}\int_{0}^{t} d\tau \frac{1}{t-\tau}\frac{q(\tau)}{c\rho}\exp\left(-\frac{\left(w+\int_{\tau}^{t} v(\eta)d\eta\right)^2 + y^2}{4k(t-\tau)} - k\alpha_n{}^2(t-\tau)\right)\phi_n(z)\phi_n(0) \qquad \text{(a.45)}$$

This modification can be obtained by directly solving the heat transfer equation (a.1).

REFERENCES

Åström, K. and J., and Björn, W., 1989, Adaptive control, Addison-Wesley.

Arpaci, V. S., 1991, Conduction heat transfer, Ginn Press.

American Welding Society, (Ed.) Conner, L. P., 1989, Welding Handbook.

Anderson, K., Cook, G. E., Karsai, G., and Ramaawamy, K., 1990, Artificial neural networks applied to arc welding process modeling and control., IEEE Transactions on industry applications, Vol. 26, No. 5, 824-830.

Burgardt, P., and Heiple, C. R., 1986, Interaction between impurities and welding variables in determining GTA weld shape. Welding Journal, June, 150-s to 155-s.

Chandel, R. S., and Bala, S. R., 1985, Cooling time and features of submerged arc welding beads. Welding Journal 64, 201-s to 208-s.

Chen., F.-C., 1990, Back-propagation neural networks for nonlinear self-tuning adaptive control., IEEE Control Systems, April, pp. 44-48.

Christensen, N., Davies, de L., and Gjermundsen, K., 1965, Distribution of temperatures in arc welding, British welding journal, February 54-75.

Chin, B. A., Madsen, N. H., and Goodling, J. S., 1983, Infrared thermography for sensing the arc welding process. Welding journal, September, 227-s to 234-s.

Committee for temperature measurement, SICE, 1971, Temperature measurement, SICE press.

Doumanidis, C.C., and Hardt, D. E. 1989, A model for in-process control of thermal properties during welding. Transactions ASME, Journal of dynamic systems, measurement and control, Vol. 111, 40-50.

Doumanidis, C. C.,1988, Modeling and control of Thermal phenomena in welding, PhD thesis, Dept. of Mechanical Engineering, Massachusetts Institute of Technology, February.

Doyle, J. C., Francis, B. A., Tannenbaum, A. R., Feedback control theory, Macmillan Publishing.

Eagar, T. W., and Tsai, N. S., 1983, Temperature fields produced by traveling distributed heat sources. Welding Journal 62, 346-s to 355-s.

Einerson, C. J., Smartt, H. B., Johnson, J. A., Taylor, P. L., and Moore, K. L., 1992, Development of an Intelligent system for cooling rate and fill control in GMAW. International Trends in Welding Science and Technology, 853-857.

Fortescue, T. R., Kershenbaum, L. S., and Ydstie, B. E., 1981, Implementation of self tuning regulators with variable forgetting factors, automatica, Vol. 17, pp. 831-835.

Franklin, G. F., Powell, J. D., and Workman, M. L., 1990, Digital control of dynamic systems, Addison Wesley.

Funahashi, K., 1989, On the approximate realization of continuous mappings by neural networks, Neural Networks, Vol. 2, pp. 303-314.

Goodwin, G. C., and Sin, K. S., 1984, Adaptive filtering prediction and control, Prentice-Hall.

Grizzle, J. W., 1986, Feedback linearization of discrete-time systems., Lecture notes in control and information sciences, 83, Springer-Verlag, p273-281.

Hardt, D. E., 1990, Modeling and control of welding process. Proc. fifth engineering foundation conference modeling of casting, welding and advanced solidification processes, TMS.

Hertz J., Krogh, A., and Palmer, R. G., 1991, Introduction to the theory of neural computation, A lecture notes volume 1 in the Santa Fe Institute studies in the sciences of complexity, Addison-Wesley.

Heiple, C. R., and Roper, J. R., 1990, The geometry of gas tungsten arc, gas metal arc, and submerged arc weld beads. Welding theory and practice, (ed.) Olson, D. L., Dixon, R., and Liby, A. L., North Holland.

Sbarbaro-Hofer, D., Neumerkel, D., and Hunt, K., 1993, Neural control of a steel rolling mill., IEEE Control Systems, June.

Inui, T., 1962, Special functions, Iwanami publishing, Japan (in Japanese).

Isidori, A., 1989, Nonlinear control systems: an introduction, Springer-Verlag.

Jakubczyk, W., 1987, Feedback linearization of discrete-time systems., System and control, Letters, 9, p411-416.

Johnson, J. A., Carlson, N. M., Smartt, H. B., and Clark, D. E., 1991, Process control of GMAW: sensing of metal transfer mode., Welding journal April, 91-s to 99-s.

Koivisto, H., Kimpimäki, Koivo, H., Neural net based control of the heating process. Artificial Neural Networks, (ed.) Kohonen, T. Mäkisara, K. Simula, O., and Kangas. J., 1991, Elesevier Science Publishers B. V. (North-Holland).

Landau, I. D., 1990, Systems identification and control design, Prentice-Hall.

Landau, I. D., 1981, Model reference adaptive controllers and stochastic self-tuning regulators, A unified approach., Transactions of the ASME, J. of Dynamic system measurement and control, Vol. 103. No. 4, pp. 404-416.

Lee, P. L., 1993, Nonlinear process control-Applications of generic model control-, Springer-Verlag.

Lee, P. L., and Sullivan, G. R., 1988, Generic model control., Computer chemical engineering, Vol. 12, No. 6, pp. 573-580.

Levin, A. U., and Narendra, K. S., 1993, Control of nonlinear dynamical systems using neural networks: Controllability and Stabilization., IEEE Transactions on neural networks, Vol. 4, No. 2, March.

Lukens, W. E., and Morris, R. A., 1982, Infrared temperature sensing of cooling rates for arc welding control. Welding Journal 61(1), pp. 27-33.

Ljung, L., 1987, System identification theory for the user., Prentice-Hall.

Marshall, J. E., 1978, Control of time-delay systems, Peter peregrinus LTD.

Masubuchi, K., 1980, Analysis of welded structures, Pergamon Press.

Moore, K. L., 1992, Iterative learning control for deterministic systems, Springer-Verlag.

Narendra, K. S., and Annaswamy, A. M., 1989, stable adaptive systems, Prentice-Hall.

Narendra, K. S., Parthasarathy, K., 1990, Identification and control of dynamical systems using neural networks., IEEE Transactions on neural networks, Vol. 1, No. 1, March, pp. 4-27.

Narendra, K. S., Parthasarathy, K., 1991, Gradient methods for the optimization of dynamical systems containing neural networks., IEEE Transactions on neural networks, Vol. 2, No. 2, March, pp. 252-262.

Nunes, A. C., 1983, An extended Rosenthal weld model. Welding Journal 62, 165-s to 170-s.

Ogata, K., 1990, Modern control engineering, Prentice-Hall.

Oreper, G. M., and Eagar, T. W., and Szekely, J., 1983, Convection in arc weld pools. Welding Journal 62, 307-s to 312-s.

Ozisic, M. N., 1968, Boundary value problems of heat conduction, International textbook company.

Psaltis, D., Sideris, A., and Yamamura, A. A., 1988, A multilayered neural network controller., IEEE Control Systems, Vol. 8, No. 2, pp. 17-21.

Press, W. H., Teukolsky, S. A., Vettering, W. T., Flannery, B. P., 1992, Numerical recipes in C: the art of scientific computing, Cambridge university press.

Rosenthal, D., 1946, The theory of moving source of heat and its application to metal treatment. Transactions ASME 43(11):849-866.

Slotine, J.-J. E., and Li, W., 1991, Nonlinear control theory, Prentice-Hall.

Song, J. B., 1992, Multivariable adaptive control in GMA welding using a thermal based depth estimator, PhD thesis, Dept. of Mechanical Engineering, Massachusetts Institute of Technology, January.

Suzuki, A., 1988, Application of adaptive control to the GTA welding process. PhD thesis, Dept. of Mechanical Engineering, Massachusetts Institute of Technology, June.

Smartt, H. B., and Einerson, C. J., 1993, A model for heat and mass input control in GMAW. Welding Journal 72, 217-s to 229-s.

Tanabe, Y., and Nakamura, H., 1981, Partial differential equations and boundary value problems, University of Tokyo press, Japan (in Japanese).

Vidyasagar, M., 1993, Nonlinear systems analysis, Prentice-Hall.

Yamakawa, T., 1993, A fuzzy inference engine in nonlinear analog mode and its application to a fuzzy logic control., IEEE Transactions on neural networks, Vol. 4, No. 3, May, pp. 496-522.

White D. A., and Sofge, D. A. (ed.), 1992, Handbook of intelligent control, neural, fuzzy, and adaptive approaches, Van Nostrand Reinhold.

Accufiber Division, Luxtron Corporation, 1990, User's manual: Accufiber model 100c, High temperature measurement and control system.