วิธีการควบคุมแบบออปติมอลที่อาศัยโครงข่ายนิวรัลสำหรับกระบวนการตกผลึกแบบกะ

นางสาวลินดา ธนะสินธนา

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมเคมี ภาควิชาวิศวกรรมเคมี
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2550

NEURAL NETWORK-BASED OPTIMAL CONTROL STRATEGY FOR A BATCH CRYSTALLIZATION

Miss Linda Thanasinthana

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering Program in Chemical Engineering
Department of Chemical Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2007

Thesis Title	NEURAL NETWORK-BASED OPTIMAL CONTROL STRATEGY FOR A BATCH CRYSTALLIZATION
By	Miss Linda Thanasinthana
Field of Study	Chemical Engineering
Thesis Advisor	Assistant Professor Amornchai Arpornwichanop, D.Eng.

Accepted by the Faculty of Engineering, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

.... Dean of the Faculty of Engineering
(Associate Professor Boonsom Lerdhirunwong, Dr.Ing.)

THESIS COMMITTEE

Paisan Kittisupakorn
.... Chairman
(Associate Professor Paisan Kittisupakorn, Ph.D.)

Amornchai Arpornwichanop
.... Thesis Advisor
(Assistant Professor Amornchai Arpornwichanop, D.Eng.)

.... External Member
(Assistant Professor Worapon Kiatkittipong , D.Eng.)

Soorathep Kheawhom
.... Member
(Soorathep Kheawhom, Ph.D.)

ลินดา ธนะสินธนา : วิธีการควบคุมแบบออปติมอลที่อาศัยโครงข่ายนิวรัลสำหรับกระบวนการตกผลึกแบบกะ. (NEURAL NETWORK-BASED OPTIMAL CONTROL STRATEGY FOR A BATCH CRYSTALLIZATION) อ. ที่ปรึกษา : ผศ. ดร. อมรชัย อาภรณ์วิชานพ, 86 หน้า.

งานวิจัยนี้เสนอการประยุกต์ใช้การควบคุมแบบออปติมอลโดยอาศัยโครงข่ายนิวรัลเพื่อควบคุมกระบวนการตกผลึกแบบกะ เนื่องจากความเข้าใจเกี่ยวกับกลไกของกระบวนการตกผลึกซึ่งเป็นกระบวนการที่มีความไม่เป็นเชิงเส้นสูงและมักแสดงพฤติกรรมที่ซับซ้อนมีค่อนข้างจำกัด ส่งผลให้การควบคุมแบบออปติมอลที่อาศัยแบบจำลองทางคณิตศาสตร์ให้สมรรถนะในการควบคุมกระบวนการได้ไม่ดีนัก นอกจากนี้ปัญหาที่พบในการควบคุมกระบวนการแบบกะคือ การวิเคราะห์คุณภาพของผลิตภัณฑ์ไม่สามารถวัดค่าแบบออนไลน์ได้ โดยจะสามารถวัดได้จากการวิเคราะห์ผลจากห้องทดลองภายหลังเสร็จสิ้นกระบวนการเท่านั้น เพื่อแก้ไขปัญหาเหล่านี้ แบบจำลองโครงข่ายนิวรัลจึงถูกพัฒนาขึ้นโดยอาศัยข้อมูลขาเข้าและขาออกของกระบวนการและนำไปใช้งานร่วมกับวิธีการควบคุมแบบออปติมอลเพื่อใช้สำหรับควบคุมเครื่องตกผลึกที่ดำเนินการแบบกะ ปัญหาการควบคุมแบบออปติมอลที่ถูกกำหนดขึ้นได้ถูกหาคำตอบโดยใช้วิธีการแบบลำดับขั้นโดยที่รูปแบบของตัวแปรควบคุมถูกแบ่งโดยใช้ฟังก์ชันแบบคงที่แบบเป็นช่วงๆ โดยนำเอาโครงข่ายนิวรัลมาประยุกต์ใช้เพื่อทำนายค่าตัวแปรโมเมนต์ซึ่งเป็นตัวแปรที่แสดงถึงคุณภาพของผลิตภัณฑ์ผลึกและทำนายค่าความเข้มข้นของสารละลายภายในเครื่องตกผลึก โดยในงานวิจัยนี้ได้เลือกกระบวนการตกผลึกแบบกะของการผลิตโพแทสเซียมซัลเฟตมาใช้เป็นกรณีศึกษาในการนำเสนอวิธีการควบคุมแบบออปติมอลโดยอาศัยโครงข่ายนิวรัล ผลการจำลองกระบวนการแสดงให้เห็นว่าการควบคุมออปติมอลโดยประยุกต์ใช้แบบจำลองโครงข่ายนิวรัลแบบใช้ซ้ำ (recursive neural network) ให้สมรรถนะการควบคุมได้ดีกว่าเมื่อเปรียบเทียบกับการวิธีการลดอุณหภูมิแบบเชิงเส้นแบบเดิม

ภาควิชา........วิศวกรรมเคมี........	ลายมือชื่อนิสิต.....ลินดา ธนะสินธนา.....
สาขาวิชา......วิศวกรรมเคมี........	ลายมือชื่ออาจารย์ที่ปรึกษา.....อมรชัย อาภรณ์วิชานพ.....
ปีการศึกษา.........2550............	

／＃ # 4970540021 : MAJOR CHEMICAL ENGINEERING

KEY WORD : NEURAL NETWORK / OPTIMAL CONTROL / BATCH CRYSTALLIZATION / SIMULATION

LINDA THANASINTHANA : NEURAL NETWORK-BASED OPTIMAL CONTROL STRATEGY FOR A BATCH CRYSTALLIZATION. THESIS ADVISOR: ASSISTANT PROFESSOR AMORNCHAI ARPORNWICHANOP, D.Eng. 86 pp.

This research presents the implementation of an optimal control with neural network predictor to control a batch crystallizer. Due to the limited understanding of nonlinear and complicated dynamics of crystallization processes, the optimal control which is a model-based control strategy may not perform well as expected. A further difficulty in batch process control is that product quality variables usually cannot be measured on-line and can only be obtained through laboratory analysis at the end of batch run. To overcome such difficulties, an artificial neural network model is developed based on input and output process data and integrated with the optimal control strategy for controlling a batch crystallizer. The formulated optimal control problem is solved by a sequential method in which the control profile is parameterized by using a piecewise constant function. The neural network is applied to predict the moment variables that represent a crystal product quality and the solution concentration within batch crystallizer. The batch crystallization of potassium sulfate production is chosen as a case study to demonstrate the proposed control strategy. The simulation results have shown that the recursive neural network-based optimal control gives a better control performance compared to a conventional linear cooling control technique.

Department……Chemical Engineering… Student's signature…Linda Thanasinthana…

Field of study…Chemical Engineering… Advisor's signature…Amornchai Arpornwichanop…

Academic year…….2007……………...

ACKNOWLEDGEMENTS

I would like to thank and express my sincere gratitude to my advisor, Assistant Professor Amornchai Arpornwichanop, for his supervision, inspiration, encouragement, advice, discussion and helpful suggestions throughout the course of this Master Degree study. Furthermore, I also grateful thank to Associate Professor Paisan Kittisupakorn as the chairman, Dr. Soorathep Kheawhom and Assistant Professor Worapon Kiatkittipong as the members of thesis committee.

Financial supports from the Thailand Research Fund (TRF-Master Research Grants) and the Department of Chemical Engineering, Chulalongkorn University are gratefully acknowledged.

I would like to thank all of my friends and colleagues in the Control and Systems Engineering Research Center for their friendship and support over the years of my study.

Special thanks go to my beloved sister Miss Apiradee Ploylearmsang, one of the most important people, and her family for their encouragement and all of the good things that they have given to me.

Finally, I would like to express the highest gratitude to my family for their love, inspiration, encouragement and financial support throughout this study.

CONTENTS

PAGE

CHAPTER

CHAPTER PAGE

CHAPTER PAGE

CHAPTER PAGE

LIST OF TABLES

LIST OF FIGURES

PAGE

PAGE

NOMENCLATURES

A	=	Total heat-transfer surface area (m^2)
b	=	An exponents relating nucleation rate to the supersaturation
B	=	Crystallization nucleation rate
c	=	Concentration of the crystallizing substance (g/g)
c'	=	Equilibrium concentration of the crystallizing (g/g)
Cm	=	Metastable concentration of the solute (g/g)
C_p	=	Heat capacity of the solution (kJ/K kg)
Cs	=	Saturation concentration of the solute (g/g)
E_b	=	Nucleation activation energy
E_g	=	Growth activation energy
g	=	An exponents relating growth rate to the supersaturation
G	=	Crystallization growth rate
k_b	=	Pre-exponential constant ((s $\mu m^3)^{-1}$)
k_g	=	Pre-exponential constant (μm/s)
k_v	=	Volumetric shape factor
M	=	Mass of solvent in the crystallizer (kg)

meanp	=	Mean of each input data
p	=	Network input data
pn	=	Normalized network input data
s	=	Supersaturation ratio
stdp	=	Standard deviation of input data
T	=	Reactor temperature
t_f	=	Final time of reaction (min)
T_j	=	Jacket temperature
U	=	Overall heat-transfer coefficient (kJ/m^{2}hK)

GREEK LETTERS

ρ	=	Density of the crystals (g/μm$^{3}{}_{)}$
μ_0	=	The zero order moment
μ_1	=	The first order moment (μm)
μ_2	=	The second order moment (μm^2)
μ_3	=	The third order moment (μm^3)
ΔH	=	Heat of reaction (kJ/kg)
μ_i^n	=	The *i*th moment of crystals formed by nucleation

μ_i^s = The *i*th moment of crystals formed by growing from seeds

ACRONYM

MSE = Mean Square Error

ANN = Artificial Neural Network

CSD = Crystals Size Distribution

CHAPTER I

INTRODUCTION

Crystallization from the solution is a widely used technique for the production of a variety of materials and chemicals that is marketed in the crystalline form. The main proposes of crystallization is to separate and produce high-purity crystal products. For this reason, the crystallization process has been widely applied to a number of industries. Generally, the operating conditions of crystallization process have a direct effect on not only the product quality, i.e., crystal purity, shape and size distribution, but also the downstream operations, i.e., filtering, drying, or formulating. The primary bottleneck to the efficient operation of manufacturing is associated with difficulties in controlling the size and shape distribution of crystals produced because of the complexity of crystallization processes. If crystal size distribution is not controlled properly, it can cause an off-specification product and difficulties in subsequence operation, i.e., long filtration or drying time.

Batch crystallizations are often used in the production of low-volume and high-value chemicals. Moreover, the operation of crystallizer in batch mode offers advantages of a narrow crystal size distribution and a large mean crystal size. In general, the final crystal size distribution of batch crystallization is affected by supersaturation, non-equilibrium driving force for crystallizations, which is a function of the crystallizer temperature. As a consequence, the control of the operating temperature at optimal conditions is crucial for obtaining a product crystal with desired quality.

In the past years, various traditional cooling methods such as linear cooling and natural cooling have been widely investigated. Recently, there has been a growing interest in the implementation of an optimal control approach to determine an optimal policy in terms of operating temperature for batch crystallizers. For examples, Miller and Rawlings (1994) implemented an open-loop optimal temperature control strategy on a bench-scale potassium nitrate-water system. With such an optimal cooling policy, the increase of the mean crystal size up to 48% compared to the natural cooling strategy was observed. Costa et al. (2005) proposed the optimal cooling temperature control to improve the performance of adipic acid process. The modeling takes into account the effect of agglomeration. The results indicated that the proposed strategy can improve the final product quality expressed in terms of the mean crystal size.

However, the effectiveness of the optimal control depends heavily on the exact knowledge of the system’s dynamic model. With the limited understanding of complex and highly nonlinear systems like crystallization processes, the optimal control do not perform as expected. One of the most effective techniques to handle with such a situation is an artificial neural network. An obvious advantage of neural network is its universal character in approximating different physical phenomena with similar computational structure. The ability to approximate complex nonlinear relationships from process data without prior knowledge of the model structure makes neural network a very attractive alternative to the classical modeling techniques (Georgieva and Azevedo, 2006).

The aim of this study is concentrated on modeling and control of a batch crystallization process by using a neural network-based technique. A neural network-based model is developed based on process data obtained from the simulation of batch crystallization for the production of potassium sulfate chosen as a case study (Shi et al., 2006). Potassium sulfate is a non-flammable white crystalline salt which is soluble in water and important chemical in fertilizer manufacture. A dynamic optimization framework is formulated using the obtained neural network model to determine an optimal operating policy of the crystallizer in terms of cooling temperature. The performance of the optimal control strategy based on the neural network model is analyzed.

1.1 Research Objective

The objective of this study is to develop an optimal control strategy based on a neural network-based model for controlling a batch crystallization process for the production of potassium sulfate.

1.2 Scopes of Research

This study implements an optimal control strategy to control a batch crystallizer. In general, the successful implementation of the optimal control relies on the accuracy of a process model. With the limited understanding of complex and highly nonlinear systems like crystallization processes, the optimal control do not perform as expected. In this study, a neural network which is an effective technique to handle with such a situation is proposed to model a batch crystallizer. Simulations of the crystallizer for the production of potassium sulfate as a case study are performed to generate input and output data to be used for training the network. The obtained neural network model is integrated with the optimal control technique to determine an optimal operating temperature policy for batch crystallization operation. The performance of the neural network-based optimal control strategy is analyzed and compared with a conventional linear cooling technique.

1.3 Research Framework

This thesis is organized as follows: First, the literature reviews related to the crystallization control and neural network are presented in Chapter II. Then, the basic theory of a crystallization process, neural network modeling and optimal control are explained in Chapter III. The modeling of batch crystallization using neural network technique is described in Chapter IV. Next, the application of the optimal control based on neural network model for controlling a batch crystallization process is presented in Chapter V. Finally, the conclusion and recommendation for future work are given in Chapter VI.

CHAPTER II

LITERATURE REVIEWS

In recent year, there has been a growing interest in the use of optimization to achieve optimal operating profiles for the control of batch crystallization. Improvement of the product quality at the end of the batch cycle is the main concern. However, the control of a batch crystallizer is a difficult and challenging problem due to their highly nonlinear and complicated dynamic behavior. An interesting way for controlling the batch crystallization is to apply the optimal control strategy with the neural network as the process model.

2.1 Crystallization Process Identification

The mathematical models of crystallization process are typically obtained through the application of population, material and energy balances and consist of a system of nonlinear partial integro-differential equations that describe the evolution of the crystal size distribution (CSD), coupled with the system of nonlinear ordinary differential equations (ODEs) that describe the evolution of the state variables of the continuous phase. There is an extensive literature on population balance modeling, numerical solution, and dynamical analysis of particulate processes. For example, Chiu and Christofides (1999) presented the approach for the development of a systematic framework for solving a number of important control problems for particulate processes, including the problem of dealing with the highly nonlinear behavior (e.g., complex growth, nucleation, agglomeration and breakage mechanisms, and the Arrhenius dependence of nucleation laws on solute concentration in crystallizers). Moreover, Chiu and Christofides (2000) presented the problem of model uncertainty and El-Farra et al. (2001) presented the problem of control under actuator constraints. Kraft (2005) reviewed the literature of nano particles and discussed the models and the numerical methods used in the small particle field. The method of moments was proposed to be the most computationally efficient approach to obtain a numerical approximation of the moments of a population balance. For this reason, this method is often used when simulating problems where transport of particles in a flow with complex geometry is essential. The method of moments has been identified as a suitable method to efficiently couple a population balance where several closure problems have to be addressed.

2.2 Control of Crystallization Process

The primary bottleneck of the crystallization process operation is the control of product qualities such as size and shape distribution of crystal. If crystal size distribution is not controlled properly, it can cause problems in the process. As a consequence, the efficient control of the process is important for obtaining a product crystal with desired quality. There are many literatures on the control of crystallization as shown in the followings.

Jones and Mullin (1974) applied optimal control theory to obtain an optimal cooling profile using method of moments for a size independent population balance model. They verified that the optimal cooling trajectory increases average crystal size as compared with linear and natural cooling strategies.

Zhang and Rohani (2003) presented an on-line optimal control methodology for the optimal quality control of a seeded batch crystallization process. In addition, an extended Kalman filter was applied to predict unmeasured state variables in the on-line optimal control system. The simulation results demonstrate that the on-line optimal control can still provide good quality control of the final product.

Hu et al. (2005) discussed the determination of optimal cooling policies for batch crystallizers that maximizing the final-time seed size and minimizing the mass ratio between the nucleated crystal and grown seed crystal. Experiments employing optimal and linear temperature profiles were performed and compared. The results indicated that the product quality of the optimal temperature profile is better than that of the linear temperature profile.

Shi et al. (2006) presented the study focusing on the development and application of predictive-based strategies for control of particle size distribution (PSD) in continuous and batch particulate processes. For batch particulate processes, the control objective was to achieve a final PSD with desired characteristics subjected to both manipulated input and product quality constraints. An optimization-based predictive control was formulated and applied to a seeded batch crystallizer. The strategy was shown to be able to reduce the total volume of the fines by 13.4% compared to a linear cooling strategy.

Nowee et al. (2007) presented a model of the cooling seeded crystallization with dissolution kinetics and identified the parameters of the kinetic sub-model. In addition, they developed dynamic optimal profiles for cooling and seed size distribution under two different objective functions and validated the model-based optimal policies experimentally. The results shown that this approach suits the problem of crystal size control very well; moreover, it paves the way for real-time optimal control.

2.3 Neural Network Designing

Artificial neural network is a mathematical structure that inspired from the attempt to simulate and understand the working of the human brain. This network involves with the learning process of the interesting system. After artificial neural network has learned what it needs to know, the trained network can be used to perform certain tasks depending on the particular application. The literatures on the designing of neural network are presented.

Mjalli et al. (2007) presented the use of artificial neural network modeling for the prediction of wastewater treatment plants performance. They showed that the reasonable results for all predicted variables were given by using multi-input networks topologies. The ANN modeling technique has many favorable features such as efficiency, generalization and simplicity, which makes it an attractive choice for modeling complex systems, such as wastewater treatment processes.

Nagy (2007) demonstrated the ability of artificial neural networks to model and control complex nonlinear biochemical processes. The simulation results were presented to demonstrate that the neural network model can achieve a good generalization. The proposed advantages of controlling by using neural network model predictive control was that it does not need a detailed knowledge about the process. In addition, it needs a very simple mathematical apparatus for the control movement calculation. This algorithm was no longer iterative thus the required computational time is very short, making it preferable for real-time applications.

Wu et al. (2007) presented the specification of input/output patterns for neural network structures that involved off-line identification algorithm. They stated that the local feed forward neural network with multiple outputs must be trained and specified in a stable operating range. For the multiple inputs, the network must be determined by a steady-state input-output map. For the multiples input/output network, the input/output patterns would induce a stable and minimum-phase neural mode.

2.4 Neural Network Control

Because of limited understanding of complex and highly nonlinear behavior of crystallization processes, the optimal control may not perform expectedly. However, application of an artificial neural network in the control strategy can solve this problem. The ability to approximate complex nonlinear relationships from process data without prior knowledge of the model structure makes the neural network a very attractive alternative to the classical modeling techniques. The literatures on the control of process by applying the neural network are shown below.

Chaudhuri and Modak presented the application of feedforward neural network for determining of the optimal substrate feed rate for fed-batch fermentation processes. The development of neural network model of the fed-batch fermentation process is described by the discrete time domain. The simulation results shown that the neural network model captures the essential features of the process kinetics, in addition it can be used for dynamic simulation results. Moreover, they proposed the recursively used of neural network model in term of predict the future value of state variables. The results showed the advantage of application of this approach that it can be easily adopted for other dynamic optimization problems such as determining temperature profile in batch reactor and multivariable optimization problems.

Ou and Rhinehart (2003) presented the model comprising of a group of sub-models that each model provides the prediction of one process output at one selected future point in time. The sub-models are mutually independent and, therefore, can run in parallel. The neural networks were used for each sub-model; in addition the prediction model was termed as a grouped neural network. The implementation of neural network based on the group neural network on a nonlinear and multivariable constrained pilot-scale distillation unit was demonstrated. The results showed that this control strategy give an effectiveness of approach for a nonlinear process subject to a variety of constraints and environmental effects.

Zhang (2005) presented the techniques for the modeling and optimization control of batch polymerization reactor. The bootstrap aggregated neural networks were applied to overcome the disadvantages of a single neural network. It was shown that the bootstrap aggregated neural networks can be effectively used in building models for batch processes. The neural network model-based iterative learning control strategy was used to utilize for the repetitive nature of batch processes. It was demonstrated that this control strategy can effectively overcome the problems due to model-plants mismatches and unknown disturbances.

Åkesson and Toivonen (2006) presented the optimal neural network control of constrained nonlinear systems. The neural network controller was designed by minimizing an MPC type cost function off-line for a set of training data. The main limitation of the neural network controller is that substantial off-line computations may be needed in order to train it properly and may not even be feasible to achieve satisfactory accuracies. Numerical examples of this work showed ability of the neural network model predictive controller in term of achieving the near-optimal control performance when compared to the optimal MPC strategy.

Georgieva and de Azevedo (2006) presented the process model of a fed-batch sugar crystallization process applying with neural networks. They presented the application of ANNs for modeling and controlling. The modeling part shown the advantages of applying ANNs with both analytical and pure data based process models. In control part, two ANN-based control algorithms were studied - model predictive control (MPC) and feedback linearizing control (FLC). The control results shown that the ANN models can capture the nonlinear process nature and give the potential advantage. However, the MPC required an online numerical optimization while the FLC required an analytical control solution.

Shomchoam (2006) presented the on-line optimal control with neural network estimator applied for the ethanol production process in a fed-batch reactor. The objective was to maximize the amount of the desired ethanol product at the end of the operation. The solution of the optimal control problem is computed using a sequential model solution and optimization method. The simulation results demonstrate that the on-line optimal control with neural network estimator can improve the control performance compared with the optimal control strategy.

CHAPTER III

THEORY

In this chapter, the theoretical background of a crystallization process, artificial neural network, modeling of batch crystallizer, and optimal control are described.

3.1 Crystallization

Crystallization is the formation of solid particles within a homogeneous phase. It may occur as the formation of solid particles in vapor, as in snow; as solidification from liquid solution, as in the manufacturing of large single crystals; or as crystallization from liquid solution (McCabe et al., 2001). Here, only crystallization from liquid solution will be described.

The crystallization process is taken place in two steps: nucleation of the crystal and crystal growth. The driving force of crystallization process is supersaturation. The terminologies involved in the crystallization process are shown below.

3.1.1 Supersaturation

The fundamental driving force for supersaturation is the difference in chemical potential between the crystallization substances in the solid and liquid phases, but the common engineering practice is to use supersaturation as the driving force of concentration.

Supersaturation is usually expressed as the supersaturation ratio

$$\mathbf{S} = \frac{\mathbf{c}}{\mathbf{c'}} \tag{3.1}$$

or the concentration difference

$$\Delta \mathbf{c} = \mathbf{c} - \mathbf{c}' \tag{3.2}$$

where c = concentration of the crystallizing substance in the solution. c’= equilibrium concentration (solubility) of the crystallizing substance in the solution

The difference in concentration may either be a difference in molar concentration of solute or a difference in molar fraction of solute.

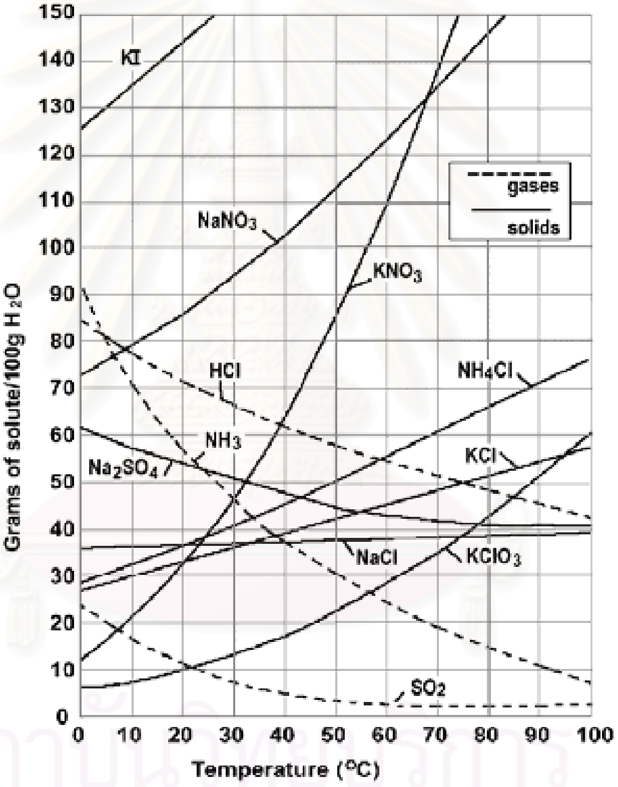


Figure 3.1 Solubility curve

When the solution is saturated, the crystallization processes will reach the equilibrium. The equilibrium concentration of solute substances in a solvent is solubility. The solubility of any substances at a given temperature is obtained experimentally by measuring the maximum amount of them that is soluble. The equilibrium relationship of the concentration of solute substances and temperature is shown by the solubility curve. The solubility of extremely small crystals is greater than that of crystals of ordinary size. In most cases, the solubility, c’, of a solute increases with the increase of temperature, but there are a few exception of this rule. A few substances have a little change in solubility with temperature, such as NaCl, others have inverted solubility curve that is their solubility decrease as the increase of temperature. The solubility curve of some substances is shown in Figure 3.1.

Supersaturation can be generated by one or more of three methods (McCabe et al., 2001). That is cooling method, evaporation and addition of a third component.

(i) Cooling or temperature reducing crystallization

If the solubility of the solution increases with the increase of temperature, as in case with many common inorganic salts or organic substances, the solution becomes supersaturated by simple cooling or reducing temperature.

(ii) Evaporative crystallization

If the solubility is relatively independent of temperature, as is the case of NaCl, supersaturation may be generated by evaporation of a portion of the solvent.

(iii) The third is addition of a third component.

If neither cooling nor evaporating is desirable, as when the solubility is very high, the supersaturation may be generated by adding the third component. The third component may act physically by mixing with the original solvent, and then the mixed solvent becomes lower solubility and the solution is become supersaturated. This process is called salting. On the other hand, if a nearly complete precipitation is required, a new solute may be created chemically by adding a third component that will react with the original solute to form an insoluble substance. This process is called precipitation. By adding the third component, the rapid creation of very large supersaturation is possible.

3.1.2 Nucleation

Nucleation of the crystals is the creation of the hypothetical solid particles in the solution. This includes the formation of small crystals nuclei in which there is or there is not a presence of the other crystals. The nucleation in case of no presence of other crystals is called primary nucleation. On the other hand, the nucleation in case of the presence of an influence of the existing macroscopic crystals in the solution is called secondary nucleation. Total nucleation is the summation of the effects of the primary and the secondary nucleation.

The primary nucleation can occur in two conditions. First is homogeneous nucleation, which is nucleation that is no influence of any other solids such as the wall of the crystallizer or particles of any foreign substances. Second is heterogeneous nucleation. This occurs when solid particles of foreign substances do an influence on the nucleation process by catalyzing and increasing the nucleation rate. For the primary nucleation, the homogeneous nucleation is rarely occurs in practice due to the high energy necessary to begin nucleation without a solid surface to catalyze the nucleation, except perhaps in some precipitation reaction.

In secondary nucleation, the two kinds of secondary nucleation are known, a fluid-shear nucleation and a contact nucleation. The fluid-shear nucleation is take place when a growing crystal is swept away at the surface by the supersaturated solution. The swept-away-nuclei become grow to a new crystal. The contact nucleation is the result of collisions between existing crystals with one another or with the walls of the crystallizer and rotary agitators. This occurs at low supersaturation where the growth rate of the crystals is at the optimum for good quality. Contact nucleation has been found to be the most effective and common method for nucleation because low energy is required and easy control without unstable operation.

There is another method of crystals nucleation that to be avoided. This method is called spurious nucleation. The crystals are attrition, more likely to crushing than to real nucleation. This happened when crystals are impacted with the moving part of the crystallizer. The sources of crystals generating are presented in Figure 3.2.

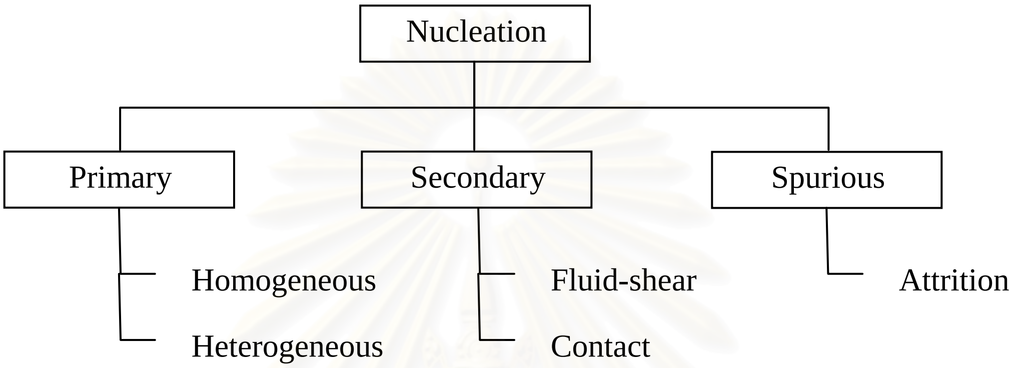


Figure 3.2 Classification of the sources of the nucleation

3.1.3 Crystal Growth

Crystal Growth is the growth of a small crystal to a macroscopic size. Solute molecules or ions in the solution reach the growing faces of a crystal by diffusion through the liquid layer. The pattern of growth resembles the rings of an onion, as shown in the Figure 3.3. The growth rate is expressed in ratio of mass of crystallizing substance per the unit area per time, and is a constant specific to the process. The growth rate is influenced by several physical factors, such as surface tension of solution, pressure, temperature, relative crystal velocity in the solution, Reynolds number, and so forth.

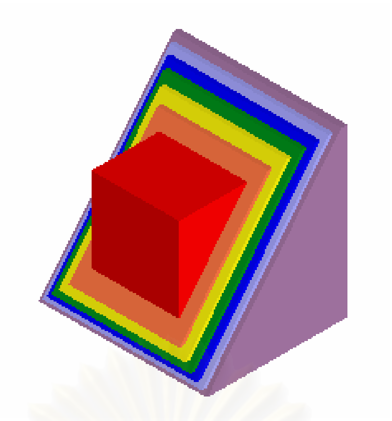


Figure 3.3 The pattern of growth of the crystal

3.1.4 Metastable Zone

The concept of supersolubility and the existence of the metastable zone are useful in understanding the behavior of the crystallization process. The metastable zone specifies the default region for operation of an industrial crystallization process to avoid uncontrolled nucleation. Figure 3.4 shows a temperature-solubility plot for a typical salt, that the solubility is a function of temperature. The plot is divided into three zones. The first is stable under saturated zone, where the crystallization is not possible because of no driving force. The second is metastable zone that is between the solubility curve and the supersolubility curve, where spontaneous crystallization is not possible. The last is labile (supersaturated) zone, where spontaneous crystallization is possible.

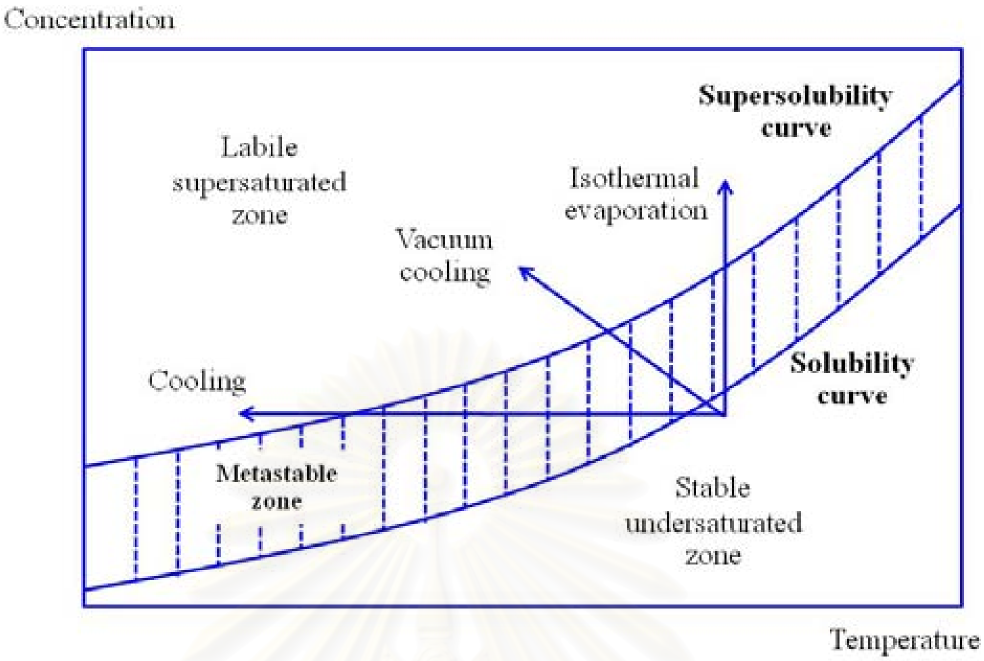


Figure 3.4 A temperature-solubility plot for a typical salt

3.1.5 Crystal Size Distribution

The appearance and size range of a crystalline product is extremely important in crystallization. If further processing of the crystals is desired, large crystals with uniform size are important for washing, filtering, transportation, and storage. The importance lies in the fact that large crystals are easier to filter out of a solution than small crystals. Also, larger crystals have a smaller surface area to volume ratio, leading to a higher purity. This higher purity is due to less retention of mother liquor which contains impurities, and a smaller loss of yield when the crystals are washed to remove the mother liquor. The theoretical crystal size distribution can be estimated as a function of operating conditions with a fairly complicated mathematical process called population balance.

3.2 Artificial Neural Network

Artificial Neural Network is the mathematical structure that emulates the functions of the biological nervous system of the human brain. The network function is determined by the connections between elements. The artificial neural network is able to learn from its environmental and adapt to its manner by adjusting the values of connections between the elements.

3.2.1 Revolution of an Artificial Neural Network

Human Nervous System is a specialized network that composes of the principle components called neurons. These neurons are interconnected to each other in complex arrangements to transmit the information between the brain and receptors. The human brain consists of about 10^{11} neurons. The shape and size of neuron is wide variety depends on its function, however, it’s typically composed of three parts: a cell body, a dendrite tree an axon as shown in Figure 3.5. The Axon is relatively thick and very long cable-like part. Its function is to send out signals to other neurons. The dendrite is thin, moderately long and branch-like part. Their function is to pick up signals sent from other neuron's axons. The cell body or soma cell is a big central part of a neuron. It contains the cell's nucleus, DNA, and all the mechanisms cells need to live and work. Like dendrites, the cell body can receive signals from axons that are in contact with it.

The development of neural network is inspired from the basic concepts of biological that mentioned above. The first artificial neuron was presented in 1943 by Warren McCulloch, the neurologist, and Walter Pits, the mathematician. They proposed the model of a simple neuron which simulates the learning algorithm of nervous cells called Habbian Learning by applying the simple mathematical equation.

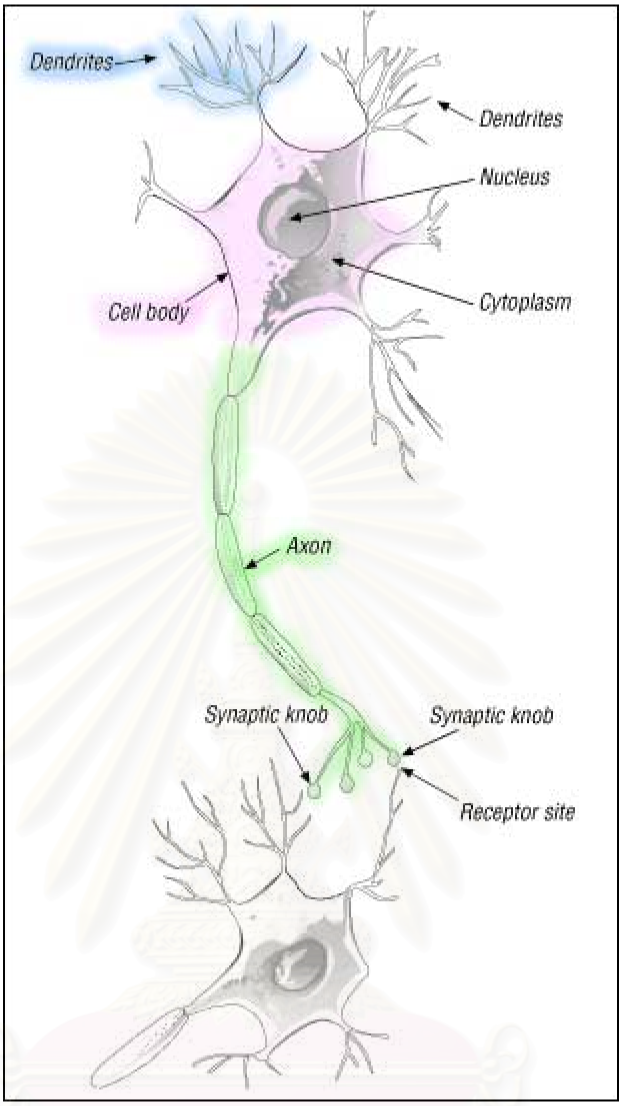


Figure 3.5 Typical biological neuron

3.2.2 Components of Artificial Neural Network

Each connection of neural nodes consists of many components for creating the artificial neural network. These components are described below.

3.2.2.1 Weighting Factors

Weight factors are the coefficients that determine the intensity of the network inputs. Each network input may have its own weighting factor. The more important inputs lead to the greater effects to the process so their weighting factors should be greater than others that less important. These weighting factors can be modified corresponding to various training sets and according to a network's specific topology.

3.2.2.2 Summation Function or Basis Function

After the network has received inputs, the inputs and their corresponding weighting factors must be combined and summed up before passing to the next step. There are two common forms of summation function, linear basis function and radial basis function.

(i) Linear Basis Function (LBF)

Linear basis function is the simplest summation function which found by multiplying each input and corresponding weight and adding up all the products. While (x_1, x_2, …, x_n) represent the input vectors and (w_1, w_2, …, w_n) represent the corresponding weight vectors. The value of the summation is shown below.

$$sum = \sum_{j=1}^{n} w_{ij} x_i \qquad (3.3)$$

(ii) Radial Basis Function (RBF)

Radial basis function is more complex than linear basis function. The function of the net value is shown below.

$$sum = \sqrt{\sum_{j=1}^{n} (x_i - w_{ij})^2} \qquad (3.4)$$

3.2.2.3 Transfer Functions or Activated Function

The results from summation function are transformed to the network output by transfer function. The transfer function can be a linear or a nonlinear function, but it is generally a non-linear function. Linear functions are not useful because the linear transfer functions are limited such as the output is simply proportional to the input (Shomchoam, 2006). For example, the most common transfer functions which are the Step, Ramp, Linear, Log-Sigmoid, Tangent-Sigmoid, Gaussian and Arc tangent transfer function are shown in Table 3.1.

Table 3.1 Transfer functions of an artificial neural network

Transfer function	Equation	Characteristic
Hard limit transfer function	$f(x) = \begin{cases} 1 \text{ if } x > 0 \\ 0 \text{ otherwise} \end{cases}$	Linear
Ramp transfer function	$f(x) = \begin{cases} 1 \text{ if } x \geq 1 \\ x \text{ if } x < 1 \\ -1 \text{ if } x < -1 \end{cases}$	Linear
Linear transfer function	$f(x) = x$	Linear
Log-Sigmoid transfer function	$f(x) = \frac{1}{1+e^{-x}}$	Non-linear
Tangent-Sigmoid transfer function	$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$	Non-linear
Gaussian transfer function	$f(x) = ce^{-x^2}$	Non-linear
Arc tangent transfer function	$f(x) = \arctan(\frac{x}{3.1416}) + 0.5$	Non-linear

Three of the most commonly used functions are hard limit transfer function, linear transfer function and log-sigmoid transfer function. One of the most commonly used functions is the log-sigmoid transfer function.

3.2.2.4 Scaling and Limiting

The scaling of the data is the normalization of the data to the range that proper to the chosen activation function. For example, normalization of the data to have means of 0 and standard deviation of 1.The algorithm of the normalization was presented in Equation (3.5).

$$pn = (p - meanp)/stdp \qquad (3.5)$$

while pn is matrix of normalized input data

p is matrix of input data

meanp is mean of each input data

stdp is standard deviation of input data

3.2.2.5 Output Function

The process of network inputs through summation function and transfer function lead to the network outputs. The network outputs must be corresponded to the chosen activation functions.

3.2.2.6 Error Function

In network training, the difference between network output values and desired output values must be calculated. This value is also called an error. The error is transformed to an error function that has an effect to the network structure. Then it will be sent to the learning function and calculating for the minimum error. The examples of common error functions are described as the follow.

(i)

Sum Square Error

$$SSE = \sum_{i=1}^{N} (y_i - p_i)^2 \qquad (3.6)$$

(ii) Mean Square Error

$$MSE = \frac{1}{N}\sum_{i=1}^{N} (y_i - p_i)^2 \qquad (3.7)$$

(iii) Mean Absolute Error

$$MAE = \frac{1}{N}\sum_{i=1}^{N} \left| y_i - p_i \right| \qquad (3.8)$$

which y_i is the network output

P_i is the network target

3.2.2.7 Learning Function

The purpose of learning function is to adjust the values of weighting factor that connect the inputs of each process element for achieving the desired results. There are two steps of learning algorithm that are supervised neural network and unsupervised neural network. The supervised neural network needs a teacher for grading the performance of the results such as training data set or the observer. In the other hand, the unsupervised neural network doesn’t need a teacher; the system must organize itself by using only output data.

3.2.3 Simple Neuron

The scalar input *p* is transmitted through a connection that is multiplied by the scalar weight *w*, to form the scalar product *wp*. The scalar product *wp* will be sent to the summer. The other input, 1, is multiplied by the bias *b* and passed to the summer. The bias is much like a weight, except that it has a constant input of 1. The transfer function net input *n* (the summer output) is the sum of the weighted input *wp* and the bias *b*. This sum is the argument of the transfer function *f*. Here *f* is a transfer function, typically a step function or a sigmoid function, that takes the argument *n* and produces the output *a*. The neuron output is calculated as *a* = *f* (*wp*+*b*). The simple neural network is shown in the Figure 3.6.

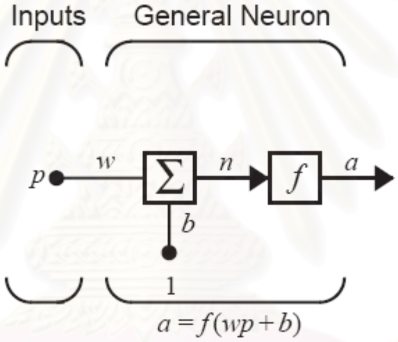


Figure 3.6 Simple neural network

Note that *w* and *b* are both adjustable scalar parameters of the neuron. Typically the transfer function is chosen by the designer, and then the parameters *w* and *b* are adjusted by the learning rule (Hagan et al., 2002). The central idea of neural networks is that such parameters can be adjusted so that the network exhibits some desired or interesting behavior. Thus, we can train the network to do a particular job by adjusting the weight or bias parameters, or perhaps the network itself will adjust these parameters to achieve some desired end.

The transfer function *f* in Figure 3.6 may be a linear or a nonlinear function of *n*. Three of the most commonly used functions are hard limit transfer function, linear transfer function and log-sigmoid transfer function. One of the most commonly used functions is the log-sigmoid transfer function.

Hard limit transfer function graph is shown below.

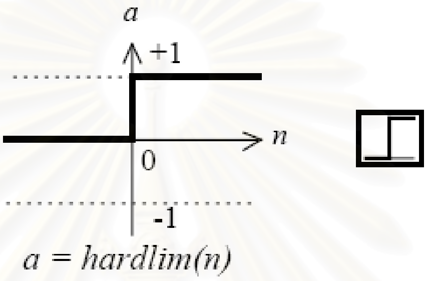


Figure 3.7 Hard limit transfer function graph

The hard limit transfer function that shown above limits the output of the neuron to either 0, if the net input argument *n* is less than 0, or 1, if n is greater than or equal to 0.

The linear transfer function is shown below.

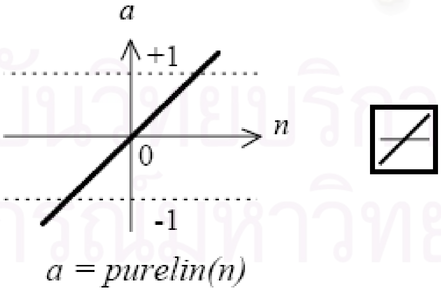


Figure 3.8 Linear transfer function

The log-sigmoid transfer function shown below takes the input, which may have any value between plus and minus infinity, and squashes the output into the range 0 to 1.

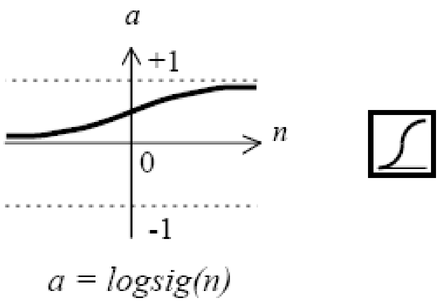


Figure 3.9 Log-sigmoid transfer function

3.2.4 Multiple-input Neural Network

Typically, a neuron has more than one input. A neuron with R inputs is shown in Figure 3.10. The individual inputs $p_1,p_2,\ldots,p_R$ are each weighted by corresponding elements $w_{1,1},w_{1,2},\ldots w_{1,R}$ of the weight matrix **W** (Hagan et al.,2002).

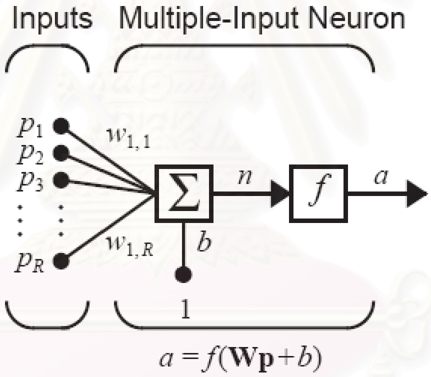


Figure 3.10 Multiple-input neural network

The neuron has a bias, which is summed with the weighted inputs to form the net input:

$$n = w_{1,1}p_1 + w_{1,2}p_2 + \ldots. + w_{1,R}p_R + b \qquad (3.9)$$

This expression can be written in matrix form:

$$n = \mathbf{Wp} + b \qquad (3.10)$$

where the matrix for the single neuron case has only one row. Now the neuron output can be written as:

$$a = f\,(\mathbf{Wp} + b) \qquad (3.11)$$

The neuron in matrix form is shown in figure 3.11

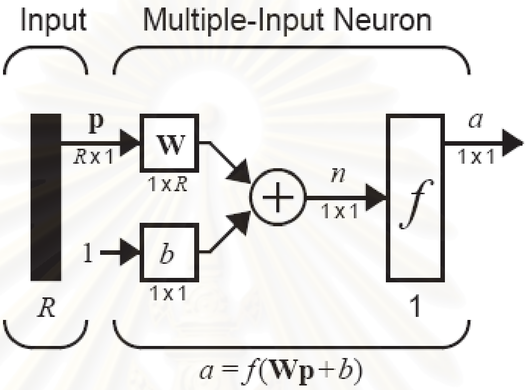


Figure 3.11 Neural network with R inputs, Matrix Notation

3.2.5 Network Architectures

Two or more of the neurons shown above may be combined in a layer, and a particular network might contain one or more such layers. First consider a single layer of neurons.

3.2.5.1 Single Layer of Neuron

Commonly one neuron, even with many inputs, is not sufficient. We might need two or more neuron to operate in parallel, in what is called a layer. A single-layer network of S neurons is shown in Figure 3.12. Note that each of the R inputs is connected to each of the neurons and that the weight matrix now has S rows. The layer includes the weight matrix **W**, the summers, the bias vector **b**, the transfer function boxes and the output vector **a**. The expression of **a** is shown at the bottom of the figure. It is common for the number of inputs to a layer to be different from the number of neurons (i.e., R ≠ S). A layer is not constrained to have the number of its inputs equal to the number of its neurons.

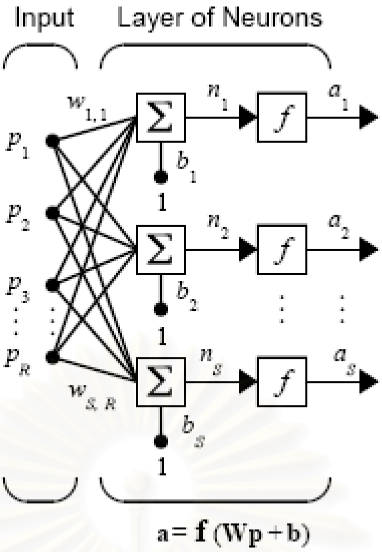


Figure 3.12 Single-layer network

3.2.5.2 Multiple Layer of Neuron

Now consider a network with several layers. Each layer has its own weight matrix **W**, its own bias vector **b**, a net input vector **n** and an output vector **a**. We need to introduce some additional notation to distinguish between these layers. We will use superscripts to identify the layers. Thus, the weight matrix for the first layer is written as $\mathbf{W}^{\mathbf{1}}$, and the weight matrix for the second layer is written as $\mathbf{W}^{\mathbf{2}}$. This notation is used in the three-layer network shown in Figure 3.13 (a) and Figure 3.13 (b). As shown, there are R inputs, S^1 neurons in the first layer, S^2 neurons in the second layer, etc. As noted, different layers can have different numbers of neurons.

Note that, the output of layer one is the input of layer two and the output of layer two is the input of layer three that is the outputs of each intermediate layer are the inputs to the following layer. Thus layer 2 can be viewed as a one-layer network with $R = S^1$ inputs, $S = S^2$ neurons, and an $S^2 \times S^1$ weight matrix $\mathbf{W}^{\mathbf{2}}$. The input to layer 2 is $\mathbf{a}^{\mathbf{1}}$, and the output is $\mathbf{a}^{\mathbf{2}}$. A layer whose output is the network output is called an output layer. The other layers are called hidden layers. The network shown in Figure 3.13 has one output layer (layer 3) and two hidden layers (layer 1 and layer 2).

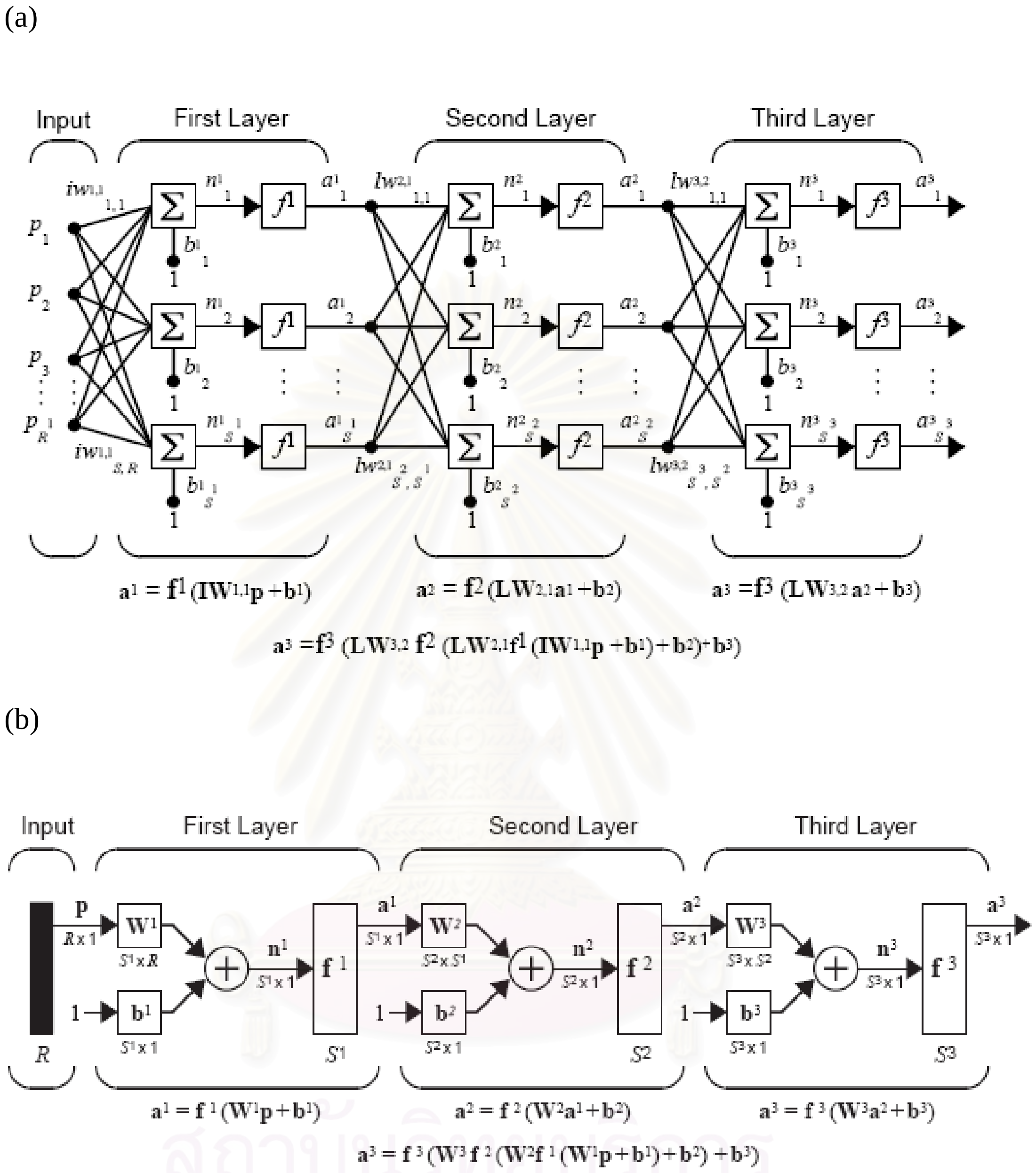


Figure 3.13 (a) Layer notation in the three layer network; **(b)** Layer notation in the three layer network, matrix notation

Figure 3.14 shows the schematic structure of multilayer feed-forward neural network providing one input layer, two hidden layers and one output layer.

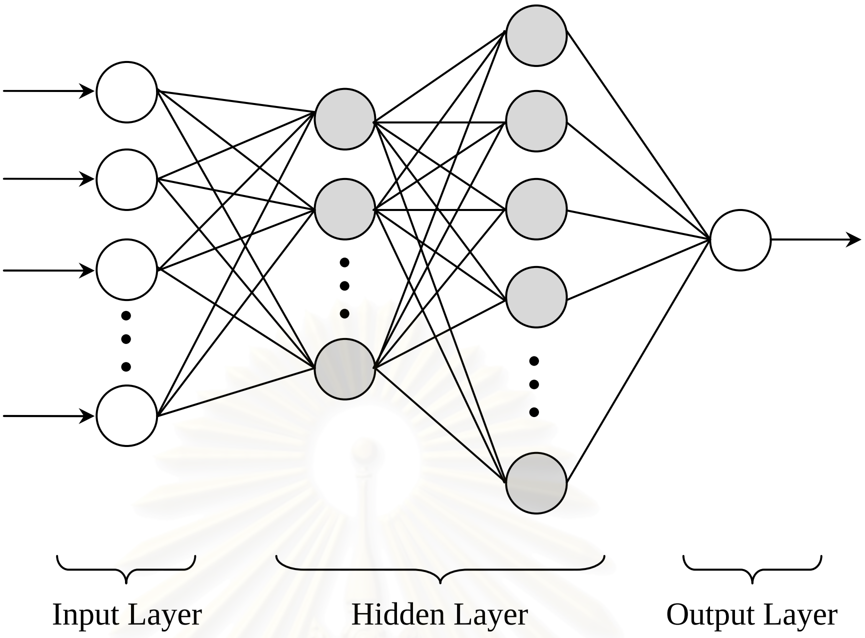


Figure 3.14 Structure of multilayer feed-forward neural network

3.3 Mathematical Model of Crystallization

3.3.1 Population Balance

The mathematical model of crystallization is constructed from material balances and energy balances of the system. The population balance equation describes the material balance that accounts for the distribution of different size crystals in crystallizer. The assumption for simplifying the equation is to consider only nucleation and growth kinetics that means do not consider on agglomeration, dendritic growth, or shape changes. A simplified population balance equation for a well-mixed batch process is shown below (Fujiwara et al., 2005).

$$\frac{\partial f}{\partial t}+\sum_{j=1}^{n}\frac{\partial}{\partial r_j}(G_j(r_j,S,T;\theta_g)f)=B(f,S,T;\theta_b)\prod_{j=1}^{n}\delta(r_j) \qquad (3.12)$$

where T is the temperature S is the supersaturation

f ($r_1,\ldots,r_n$,t) is the crystal size distribution

r_i is the *i*th characteristic growth dimension

G*i* = dr*i*/dt is the growth rate along r*i*

B is the nucleation rate which is typically some integral function of *f*

δ is the Dirac delta function

θ_g is a vector of growth kinetic parameters

θ_b is a vector of nucleation kinetic parameters

3.3.2 Moment Model

Although software is available for simulating the population balance as mentioned earlier, a simplified moment model is typically used for identifying the kinetics parameters. This model is obtained by multiplying both sides of a population balance equation by power of r_i and integrating. For crystals with one characteristic growth dimension and size-independent growth, these moment equations are as follows (Fujiwara et al., 2005):

$$\frac{d\mu_0}{dt} = B(\mu_k, S, T; \theta_b) \tag{3.13}$$

$$\frac{d\mu_j}{dt} = iG(S, T; \theta_g)\mu_{j-1} \qquad ;\ \mathrm{j} = 1,2,\ldots \tag{3.14}$$

where the moments are

$$\mu_j(t) = \int_0^{\alpha} r^i f(r,t)dt \tag{3.15}$$

The moments are related to physical properties of interest such as crystal number, length, area, and volume. The value of k is typically 2 or 3. The number of moments needed to describe the crystallization process depends on the nucleation mechanism that is dominant in the crystallizer. Assumptions are typically made so that the moment equations are closed, that is, there exists an integer j such that derivatives of the lower order moments do not depend on the higher order moments. Hence a small number of sparse ordinary differential equations are solved instead of the partial differential equation (the simplified population balance equation). The model is completed by a material balance on the solute and energy balance for the multiphase system (Fujiwara et al., 2005).

3.4 Optimal Control

Optimal control, also known as a dynamic optimization problem, involves determining a control profile for a dynamic system that optimizes a given performance index. The dynamic system is usually represented by sets of differential and algebraic equations (DAEs) derived from dynamic mass and energy balances, and physical and thermodynamic relations.

A general dynamic optimization problem can be stated as follows:

Find u(t) over t ∈ [t_0, t_f] maximizing or minimizing J:

$$J = \theta[x(t_f)] + \int_{t0}^{tf} \phi[x(t),u(t),t]dt \tag{3.16}$$

Subject to

$$\frac{dx}{dt} = f[x(t),u(t),t] \tag{3.17}$$

$$x(t_0) = x_0 \tag{3.18}$$

$$h[x(t),u(t),t] = 0 \tag{3.19}$$

$$g[x(t),u(t),t] \leq 0 \tag{3.20}$$

$$x^L \leq x(t) \leq x^U \tag{3.21}$$

$$u^L \leq u(t) \leq u^U \tag{3.22}$$

where *J* is the performance index or desired objective function, x and u are the vector of state and control variables, respectively, Equation (3.17) is the system of ordinary differential equations, Eq. (3.18) is the initial condition for Equation (3.17), Equation (3.19) and Equation (3.20) are the equality and inequality algebraic constraints, respectively, and Equation (3.21) and Equation (3.22) are the upper and lower bounds on the state and control variables, respectively.

The solution of optimal control problems have been a subject of research for many years. There are several different computational techniques available for solving the optimal control problems. The indirect methods focus on obtaining a solution to the classical necessary conditions for optimality. These methods are also known as variational methods. However, it has been found that these methods result to a two-point boundary value problem which is difficult to solve. Thus, the direct methods which transform the original optimal control problem into a finitedimensional nonlinear programming problem and solve it directly are proposed. Depending on the degree of discretization, the direct methods can be classified into two general strategies. In the simultaneous methods, the control and state variables are discretized (full discretization) whereas only the control variables are discretized (partial discretization) in the sequential methods.

3.4.1 Variation Methods

These methods are based on the solution of the first order necessary conditions for optimality that are obtained from Pontryagins's Maximum Principle (PMP). According to PMP, the problems of minimizing the objective function J in Equation (3.16) subject to dynamic constraints represented by Equation (3.17) to Equation (3.20) can be reformulated as that of minimizing the Hamiltonian function. These procedures lead to a two-point boundary value problem (TPBVP) that can be solved with different approaches, including single shooting, multiple shooting, invariant embedding or some discretization methods such as collocation on finite elements or finite differences. The limitation of these methods is the complexity in the solution of differential-algebraic equations.

3.4.2 Simultaneous Methods

In the simultaneous methods, both state and control variable profiles are discretized by approximating functions and treated as decision variables in the optimization problem. The process dynamic models and the optimization problems are solved at the same time. These avoid solving the model equations at each of iteration in the optimization algorithm as in the sequential methods. In this approach, the dynamic process model constraints in the optimal control problems are transformed to a set of algebraic equations which is treated as equality constraints in the NLP problem. As a result, the optimal control problems are reduced to a constrained nonlinear optimization problem. To solve this problem, Successive Quadratic Programming (SQP), reduced space SQP, the interior-point approach and the conjugate gradient methods can be used to solve efficiently.

The general algorithm for the simultaneous methods is as follows:

Problem:

$$\underset{x(t),u(t)}{Min}\ \Phi[x(t),u(t),t] \qquad \text{Objective function}$$

Subject to

$$\begin{array}{lll} \dfrac{dx}{dt} = f[x(t),u(t),t] & \text{Process dynamic equation} & \\ x(0) = x_0 & \text{Initial conditions for states} & (3.23) \\ h[x(t),u(t),t] = 0 & \text{Equality constraints} & \\ g[x(t),u(t),t] \le 0 & \text{Inequality constraints} & \\ x^L \le x(t) \le x^U & \text{State profile bounds} & \\ u^L \le u(t) \le u^U & \text{Control profile bounds} & \end{array}$$

Step1: Discretize the process states and inputs using any standard collocation method (e.g. orthogonal collocation).

$$x_{K+1}(t) = \sum_{i=o}^{K} x_i \phi_i(t) \quad \text{where} \quad \phi_i(t) = \prod_{k=0,i}^{K} \frac{(t-t_k)}{(t_i-t_k)} \quad , \ x_{K+1}(t_i) = x_i \qquad (3.24a)$$

$$u_K(t) = \prod_{i=1}^{k} u_i \psi_i(t) \quad \text{where} \quad \psi_i(t) = \prod_{k=1,i}^{K} \frac{(t-t_k)}{(t_i-t_k)} \quad , \ u_K(t_i) = u_i \qquad (3.24b)$$

Step2: Substitute the discrete states and inputs into process dynamic model and obtain the algebraic expression for residuals.

$$\frac{dx}{dt} = f[x(t),u(t),t] \qquad \text{for } i = 1,2,\ldots,K \qquad (3.25)$$

with $x(0) = x_0$

Step3: Substitute the discretized dynamic model into the original optimal control problem.

$$\underset{u(t)}{Min}\,\Phi[x(t),u(t),t]$$

Subject to

$$\frac{dx}{dt} \;=\; f\,[x(t)\,,u_K(t),t] \qquad \text{when} \;\; i = 1,2,...,K$$

$$u_K(t) = \prod_{i=1}^{k} u_i \psi_i(t) \qquad \text{where} \;\; \psi_i(t) = \prod_{k=1,i}^{K} \frac{(t-t_k)}{(t_i - t_k)} \;\;,\;\; u_K(t_i) = u_i$$

$$x(0) \;=\; x_0 \qquad (3.26)$$

$$h[x(t),u(t),t] \;=\; 0$$

$$g[x(t),u(t),t] \;\leq\; 0$$

$$x^L \;\leq\; x(t) \;\leq\; x^U$$

$$u^L \;\leq\; u(t) \;\leq\; u^U$$

Step4: Choose t_i by using orthogonal collocation method.

Step5: Solve problem given in Step3 at the t_i chosen in Step4 using any non-linear programming problem solver such as Successive Quadratic Programming (SQP).

3.4.3 Sequential Methods

In the sequential methods, only the control variables are discretized. These techniques are also known as control vector parameterization methods. Typically, a piecewise constant approximation over equally spaced time intervals is made for the inputs. Given the initial conditions and a given set of control parameters, the process model equations are solved with a differential-algebraic equation solver at each of iteration. This produces the value of the objective function, which is used by a nonlinear programming solver to find the optimal parameters in the control parameterization.

The general algorithm of the sequential methods is as follows:

Problem:

$$\underset{u(t)}{Min}\,\Phi[x(t),u(t),t] \qquad \text{Objective function}$$

Subject to

$$\begin{aligned}
\frac{dx}{dt} &= f[x(t),u(t),t] && \text{Process dynamic equation} \\
x(0) &= x_0 && \text{Initial conditions for states} \qquad (3.27) \\
h[x(t),u(t),t] &= 0 && \text{Equality constraints} \\
g[x(t),u(t),t] &\le 0 && \text{Inequality constraints} \\
x^L \le x(t) &\le x^U && \text{State profile bounds} \\
u^L \le u(t) &\le u^U && \text{Control profile bounds}
\end{aligned}$$

Step1: Discretize the process inputs using any standard collocation method (e.g. orthogonal collocation).

$$u_K(t) = \prod_{i=1}^{k} u_i \psi_i(t) \quad \text{where} \quad \psi_i(t) = \prod_{k=1,i}^{K} \frac{(t-t_k)}{(t_i-t_k)} \ , \ u_K(t_i) = u_i \qquad (3.28)$$

Step2: Substitute the parameterized inputs into the process dynamic model

$$\frac{dx}{dt} = f[x(t),u_K(t),t] \qquad \text{when } i = 1,2,...,K \qquad (3.29)$$

with $x(0) = x_0$

Step3: Substitute the modified process dynamic model given by Equation (3.29) into the problem given by Equation (3.27). The updated problem statement according to sequential methods is given by Equation (3.30).

$$\underset{u(t)}{Min}\,\Phi[x(t),u(t),t]$$

Subject to

$$\frac{dx}{dt} = f[x(t),u_K(t),t] \qquad \text{when } i = 1,2,...,K$$

$$u_K(t) = \prod_{i=1}^{k} u_i \psi_i(t) \quad \text{where} \quad \psi_i(t) = \prod_{k=1,i}^{K} \frac{(t-t_k)}{(t_i - t_k)} \ , \ u_K(t_i) = u_i$$

$$x(0) \ = \ x_0 \tag{3.30}$$

$$h[x(t),u(t),t] \ = \ 0$$

$$g[x(t),u(t),t] \ \leq \ 0$$

$$x^L \ \leq \ x(t) \ \leq \ x^U$$

$$u^L \ \leq \ u(t) \ \leq \ u^U$$

Step4: Choose t_i using orthogonal collocation method and evaluate u as a function of time by using Equation (3.28).

Step5: Choose initial guess for decision variables and solve the dynamic process model given by Equation (3.29) in Step2 for the input obtained in step4 using any ODE solver.

Step6: Evaluate the objective function given in Equation (3.30) using state and control profiles obtained in step5 and update the values of decision variables using any standard optimization routine such as steepest descent or Quasi- Newton methods. Repeat step4 through step6 until convergence.

The main advantage of the sequential methods is that only the control profiles are discretized and considered as the decision variables. The optimization formulated by this approach is a small scale nonlinear programming problem. However, the limitation of these methods is a difficulty to handle a constraint on state variables (path constraints). This is because the state variables are not directly included in the nonlinear programming problem.

CHAPTER IV

NEURAL NETWORK MODELING OF BATCH POTASSIUM SULFATE CRYSTALLIZATION

Agriculture has played an important role in the human life. Agricultural products are, for example, wheat, corn, rice, soybeans, or milk. Many governments in the world have subsidized the agriculture to ensure an adequate food supply. Supporting of the development of fertilizers is one of many ways that they provided. Fertilizers are the compounds that is given to plants to promote growth. They are usually applied either through the soil which is uptake by plant roots, or by foliar feeding which is uptake through leaves. Fertilizers can be organic or inorganic. The organic fertilizers are composed of organic matter such as manure, slurry, worm castings, peat, seaweed, sewage or guano. In the other hand, the inorganic fertilizers are made of simple or inorganic chemicals or minerals.

Although the advantages of organic fertilizers are low cost, they can have many disadvantages. The density of nutrients in organic fertilizers is comparatively modest and the composition of organic fertilizers tends to be more complex and variable than a standardized inorganic products. Improper processed organic fertilizers may contain pathogens from plant or animal matter that are harmful to humans or plants. In non-organic farming a compromise between the use of artificial and organic fertilizers is common, often using inorganic fertilizers supplemented with the application of organics that are readily available such as the return of crop residues or the application of manure. Advantages of inorganic fertilizers are that they provide an accurate amount of nutrients. For this reasons, the inorganic fertilizers gain more interesting in the field of fertilizer development. One of the most interesting inorganic fertilizers is potassium sulfate.

4.1 Mathematical Model of a Batch Crystallization Process of Potassium Sulfate Production

Potassium sulfate (K_2SO_4) is a non-flammable white crystalline salt which is soluble in water. Potassium sulfate is the world's most popular low-chloride fertilizer. It has a very low salinity index making it the preferred potash fertilizer in areas at risk from soil salinity. Potassium sulfate can improve crop yield and quality. It makes plants more resistant to drought, frost, insects and disease. Not only does potassium sulfate improve crop's nutritional value, taste and appearance but also improve its resistance to deterioration during transport and storage, and its suitability for industrial processing.

In this work, the process model for potassium sulfate production developed by Shi et al. (2006) was applied to simulation studies. The proposed mathematical model which consists of differential and algebraic equations is presented below. Figure 4.1 shows the schematic of a batch crystallizer.

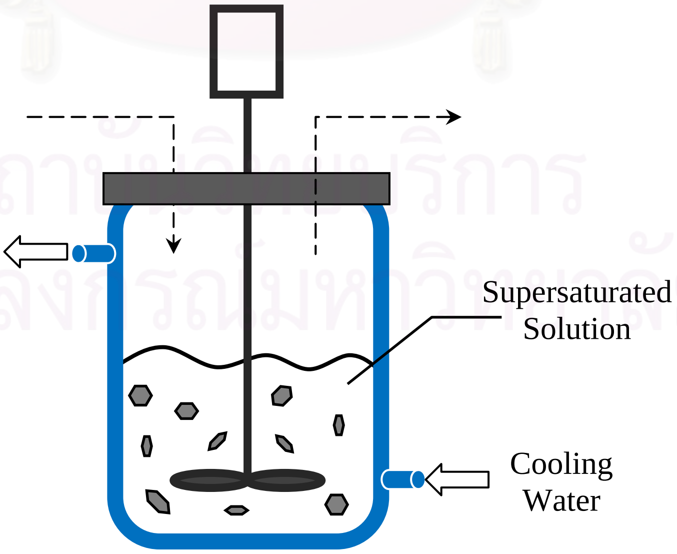


Figure 4.1 A schematic of a batch crystallizer

Population Balance:

$$\frac{\partial n(r,t)}{\partial t} + G(t)\frac{\partial n(r,t)}{\partial r} = 0 \tag{4.1}$$

Material Balance:

$$\frac{\partial C}{\partial t} = -3\rho k_v G(t)\mu_2(t) \tag{4.2}$$

Energy Balance:

$$\frac{\partial T}{\partial t} = -\frac{UA}{MC_p}(T - T_j) - \frac{\Delta H}{C_p}3\rho k_v G(t)\mu_2(t) \tag{4.3}$$

where G is crystallization growth rate, C is the solute concentration, ρ is the density of the crystals, k_v is the volumetric shape factor, T is the reactor temperature, U is the overall heat-transfer coefficient, A is the total heat-transfer surface area, M is the mass of solvent in the crystallizer, C_p is the heat capacity of the solution, T_j is the jacket temperature, ΔH is the heat of reaction.

The nucleation rate, $B(t)$, and the growth rate, $G(t)$, are given by:

$$B(t) = k_b e^{-E_b/RT}\left(\frac{C - C_S(T)}{C_S(T)}\right)^b \mu_3 \tag{4.4}$$

$$G(t) = k_g e^{-E_g/RT}\left(\frac{C - C_S(T)}{C_S(T)}\right)^g \tag{4.5}$$

where E_b is the nucleation activation energy, E_g is the growth activation energy, b is an exponent relating nucleation rate to the supersaturation, g is an exponent relating growth rate to the supersaturation, C_s is the saturation concentration of the solute.

Equations 4.6 - 4.7 show the saturation concentration (C_s) and metastable concentration (C_m). These two concentrations represent the constraints of the solution concentration corresponding to the solution temperature, i.e. $C_s \leq \mathrm{C} \leq C_m$. This condition must hold during the whole process.

$$C_s(T) = 6.29\times10^{-2} + 2.46\times10^{-3}T - 7.14\times10^{-6}\,T^2 \qquad (4.6)$$

$$C_m(T) = 7.76\times10^{-2} + 2.46\times10^{-3}T - 8.10\times10^{-6}\,T^2 \qquad (4.7)$$

For a particle population with distribution $n(r,t)$, the ith moment is defined as

$$\mu_i = \int_0^{\alpha} r^i n(r,t)dr \qquad (4.8)$$

The zero order moment (μ_0) is the total number of particles in the system. The first order moment (μ_1) is the length of the particles in the system. The second order moment (μ_2) is the area of the particles in the system. The third order moment (μ_3) is the volume of the particles in the system.

Because the nucleation and growth rates are assumed to be independent on particle size, this allows the approach of the moments equations, which results in the reduced-order moments model being an exact replication of the evolution of the dominant modes of the PBM. The differences between the two methods are only due to numerical errors in the integration of two time-varying systems. It is noted that the method of moments has been extensively used in the past to analyze the dynamics of particulate processes.

By applying this moment operator to the population balance equation, a set of moment equations can be developed with the general form. Equations 4.9 – 4.11 show the model of first four moments of the particle size distribution of the crystals.

$$\frac{d\mu_i^n}{dt} = iG(t)\mu_{i-1}^n(t), \qquad i=1,2,3 \qquad (4.9)$$

$$\mu_0^s = k_4 \qquad (4.10)$$

$$\frac{d\mu_i^s}{dt} = iG(t)\mu_{i-1}^s(t), \qquad i = 1,2,3 \qquad (4.11)$$

The definitions of the crystals formed by nucleation and by growing from the seeds are shown below.

$$\mu_i^n = \int_0^{r_g} r^i n(r,t)dr \qquad i = 1,2,3 \qquad (4.12)$$

$$\mu_i^s = \int_{r_g}^{\infty} r^i n(r,t)dr \qquad i = 1,2,3 \qquad (4.13)$$

The parameters of the model are shown in Table 4.1.

Table 4.1 Parameter values for the batch crystallizer in the case study

Parameters	Values	Parameters	Values
b	1.45	g	1.5
k_b	285.01 (s μm^3)$^{-1}$	k_g	1.44 × 10^8 μm/s
Eb/R	7517.0 K	Eg/R	4859.0 K
U	1800 kJ/m^{2}hK	A	0.25 m^2
ΔH	44.5 kJ/kg	C_p	3.8 kJ/K kg
M	27.0 kg	ρ	2.66 × 10^{-12} g/μm^3
k_v	1.5	t_f	30 min

Initial conditions are as follows.

$$n(0,t) = \frac{B(t)}{G(t)} \tag{4.14}$$

The initial seed distribution of the seeded batch crystallizer is assumed to be a parabolic distribution, from 250 to 300 μm, and the maximum density of initial seed distribution, which is 2/μm g solvent, occurs at 275 μm, i.e.,

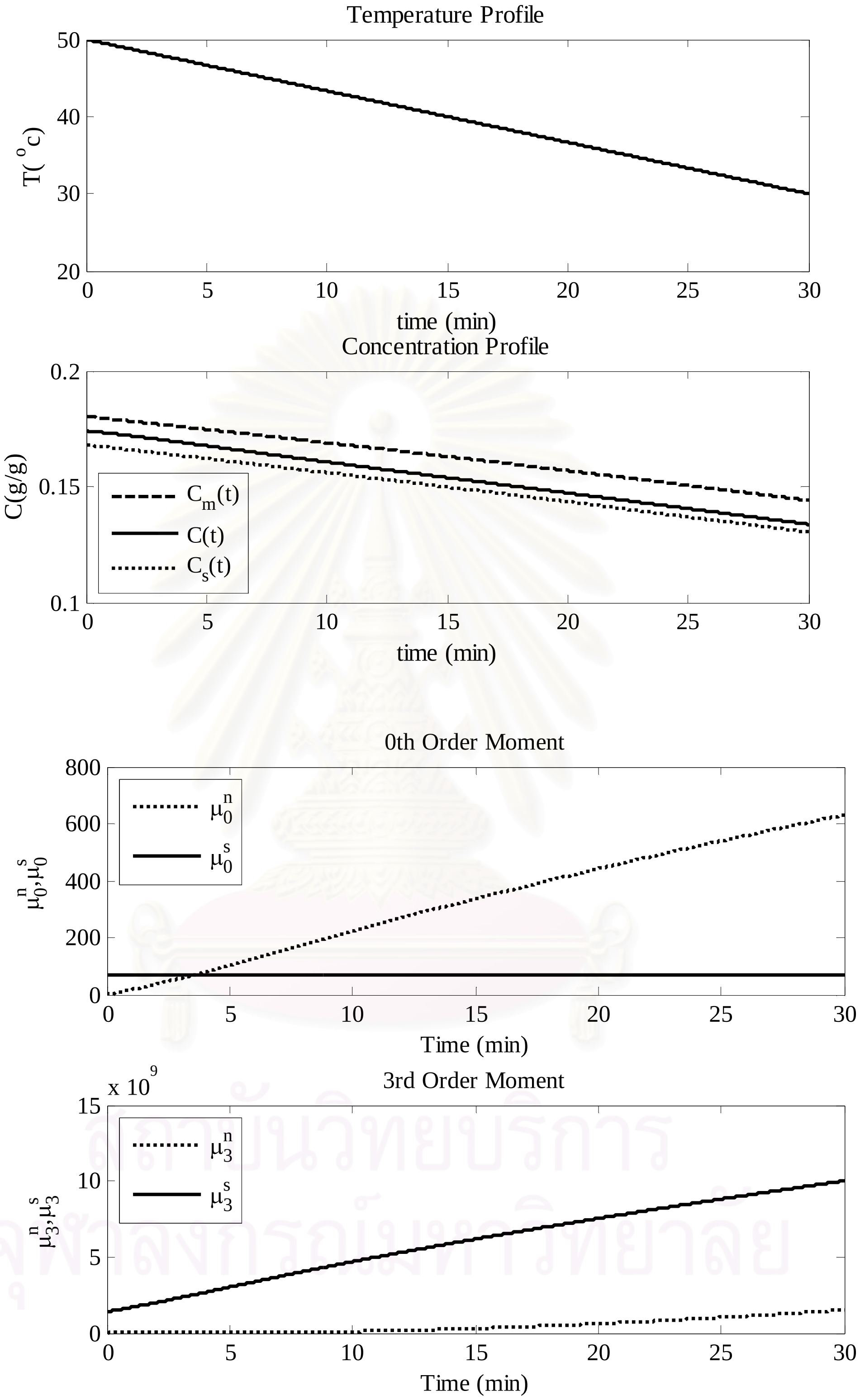
for 250 μm $\leq$ r $\leq$ 300 μm

$$\mathrm{n(r,0) = 0.0032(300-r)(r-250)} \tag{4.15}$$

for r < 250 μm and r > 300 μm

$$\mathrm{n(r,0) = 0} \tag{4.16}$$

To study the dynamic behavior of the process, the Euler method with 1500 discrete points is applied to simulate these models. The evolution of the reactor temperature, *T*, and solution concentration, *C*, is shown in Figure 4.2(a). Note that, the reactor temperature in the simulation results is decreased linearly from 50 ^{o}C to 30 ^{o}C. The dashed and dotted lines represent the upper and lower constraints on the concentration, respectively. Figure 4.2(b) shows the evolution of the zero and third moments of the PSD of the crystals formed by nucleation (dashed lines) and those growing from the seeds (solid lines).



(a)

(b)

Figure 4.2 Simulation results for the linear cooling strategy: **(a)** temperature and concentration profile; **(b)** the zero and third moments of the particle size distribution of the crystals

4.2 Neural Network Designing

In process identification, the previous process inputs and corresponding target outputs are applied to train the network until it can approximate a function. Then the trained network will be applied to represent the process. There are generally seven steps in the process modeling by using neural network.

1. Assemble the training data
2. Data processing
3. Creating the network object
4. Initializing weighting factor
5. Training the network
6. Network Validation
7. Simulating the network response to new inputs

4.2.1 Training data

The first step of network modeling is to define the process by network training. The training data consists of input data and output data. The input and output data can be collected from the operation of the plant or generated by a reliable process model. The selected input data should be the data that have an effect to the predicted output variable. In this work, the data that we want to predict is the future data of solution concentration and the moments of the particles in the system. Because both the reactor temperature and concentration of the solution are the parameters that have effects to those predicted output data, so the solution concentration and reactor temperature are chosen as the input data. However, only the present-timed data may be not enough to represent and predict the output. For this reason, the input data is consisted of 5 value: reactor temperature at time (t), (t-1) and (t-2) and concentration at time (t) and (t-1). The desired output data is concentration at time (t+1), total volume of nucleated fine particles i.e., the third moment of the crystals formed by nucleation (μ_3^n) at time (t+1) and total volume of the crystals growing from the seeds (μ_3^s) at time (t+1). The schematic input-output structure is presented in Figure 4.3.

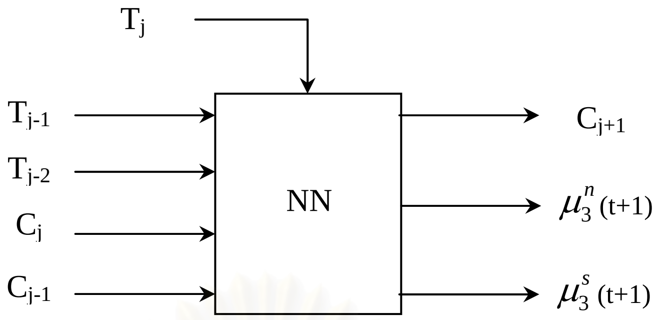


Figure 4.3 Data input-output structure of the work

These training data are obtained by simulating the mathematical models and solved in MATLAB programming. The simulation is to vary the value of the reactor temperature. The reactor temperature is decreased randomly in the range of 50°C to 30°C. Randomly decreasing temperature profiles with 10 intervals for each profile are applied. One of the random profiles with 10 is shown in figure 4.4.

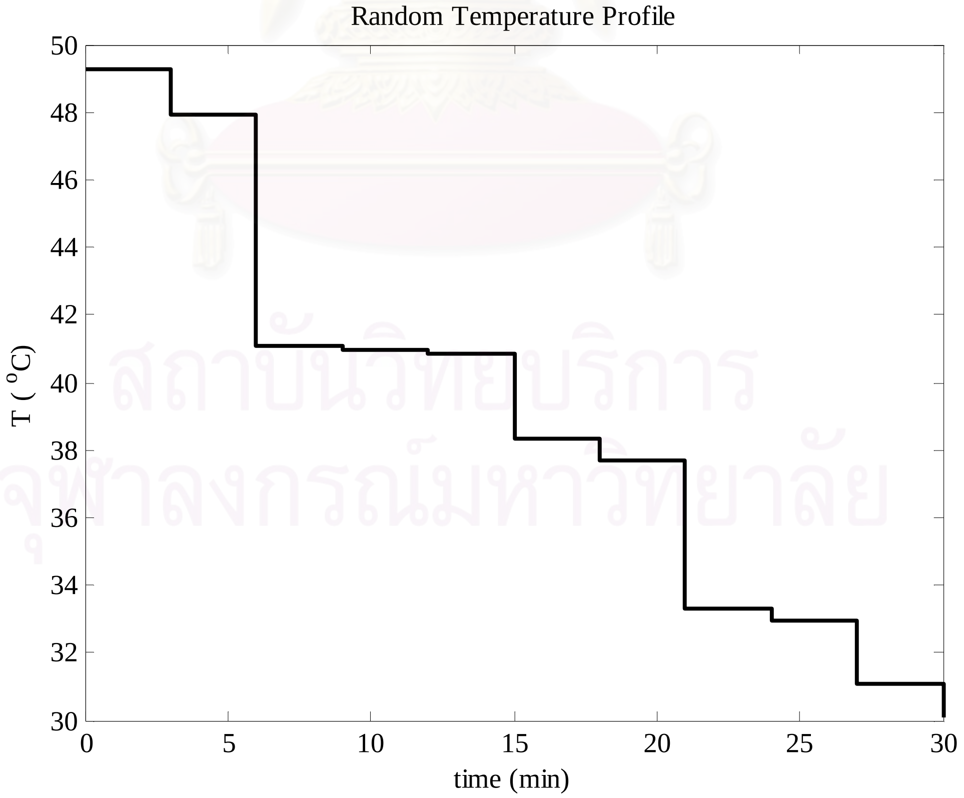


Figure 4.4 Example of randomly decreasing temperature profile

In this work, all data were normalized to have means of 0 and standard deviation of 1.The algorithm of the normalization was presented in Equation 4.17.

$$pn = (p - meanp)/stdp \tag{4.17}$$

where pn is matrix of normalized input data
p is matrix of input data
meanp is mean of each input data
stdp is standard deviation of each input data

Moreover, we should divide the data into training data, validation data, and test data before training the network. The validation data are used to validate how well the network generalized and the test data provide an independent test of network generalization to data that the network has never seen. This work, the data set is randomly divided into three sets. 60% of input data is a training data, 20% of input data is a validation data and another last 20% of input data is a test data.

4.2.2 Defining the Network

The next step is to create the network. There is no standard procedure to determine the network structure, however, the general procedure is to fix an initial size and then check the error tolerance of this structure. If this error is satisfied, the training process is stopped. If not, the size and the structure are revised and the whole procedure repeats until it satisfies the tolerance. The most common network used with backpropagation is the two-layer feed-forward neural network because it can potentially represent any input-output relationship with a finite number of discontinuities.

After create the new network, we have to define three arguments. The first argument is a matrix of input data. The second is a matrix of target or output data. These data come from the assemble data step described above. The third argument is the sizes of each hidden layer. The output layer size has not to define in this step because it was determined by the targets output data size.

More optional arguments can be provided. For example, the transfer functions of each layer or the name of the training function. If these arguments are not supplied, the program will provide the default transfer function for hidden layers as tan-sigmoid transfer function and the default for the output layer as linear transfer function.

4.2.3 Initializing weighting factor

The initial weighting factor has an effect on the speed and quality of neural network training. The weighting factors are normally initialized randomly so each connection will respond slightly differs during training. The weighting factors will be repeatedly re-initialized until the error is satisfied.

4.2.4 Training the network

Training is a procedure to determine the optimal values of the connection weights and bias weights. Once the network weights and biases have been initialized in the previous step, the network is ready for training. The training process requires a set of examples of proper network behavior that are network inputs and target outputs, which are obtained from the assemble step described before. During training the weights and biases of the network are iteratively adjusted to minimize the network performance function. There are several different training algorithms for feedforward networks. All of the algorithms use the gradient of the performance function to determine how to adjust the weights to minimize performance. The gradient is determined using a technique called backpropagation, which involves performing computations backwards through the network. The basic backpropagation training algorithm, in which the weights are moved in the direction of the negative gradient, is described in the Appendix A. The schematic algorithm of the training step is represented by Figure 4.5.

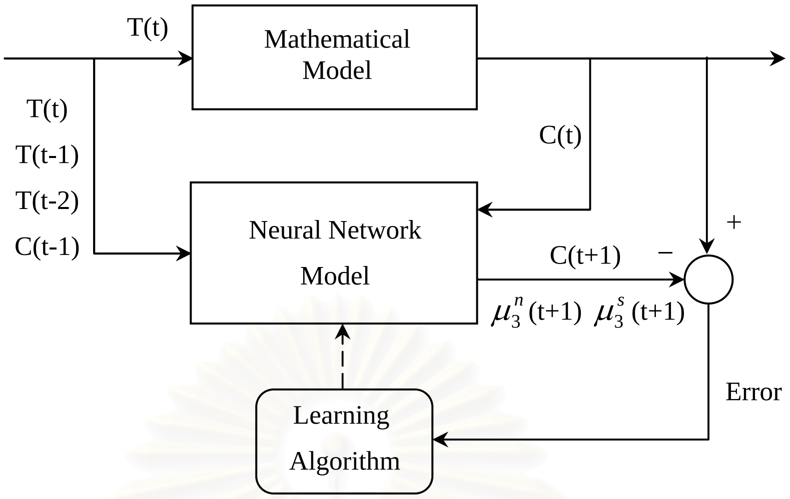


Figure 4.5 The algorithm of neural network training

In the current work, the chosen performance function for feedforward networks is mean square error (MSE) - the average squared error between the network outputs and the target outputs. The chosen training algorithm is Levenberg-Marquardt because it is the fastest method (Demuth et al., 1992-2006). The Levenberg-Marquardt algorithm is described in the Appendix B.

The MSE for the various architectures of a neural network when applied with patterns of temperature profiles is presented in Table 4.2 to Table 4.4. The architecture which gives the minimum value of the mean square error is the optimum network structure for application as a neural network model. The optimum architecture of a neural network in this work is a three-layer feed-forward neural network with 5 nodes in input layer and 3 nodes in output layer. The number of node in 1st, 2nd, and 3rd hidden layer is 3, 7, 5 nodes respectively. The hidden layer transfer function is log-sigmoid transfer function and the output layer transfer function is linear transfer function. The summation of network details is described in Table 4.5. In addition, the schematic of network structure is shown in Figure 4.6.

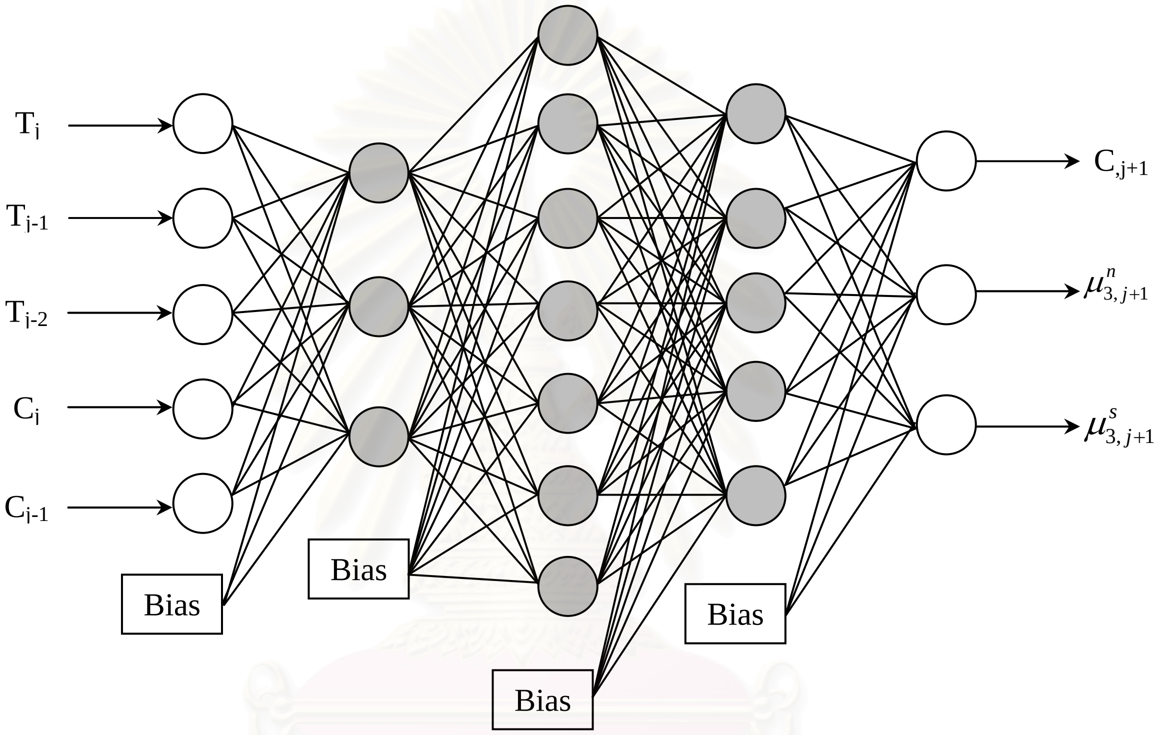


Figure 4.6 The schematic of the optimum designed network structure

Table 4.2 MSE of various network structures: 1 layer

Number of Nodes in Hidden Layer	Mean Square Error (MSE) [×10^{-4}]	
	log-sigmoid Transfer Function	tan-sigmoid Transfer Function
1	290.58	2.9058
2	8.9324	4.5401
3	6.9829	6.2904
4	18.566	1.9773
5	6.1386	6.4649
6	6.1956	6.2918
7	6.2663	6.0240
8	6.1564	5.6754
9	5.7161	5.6329
10	6.1478	6.5369
11	4.6616	6.3733
12	5.7908	6.4524
13	5.3304	5.7603
14	5.5640	5.9805

Table 4.3 MSE of various network structures: 2 layers

Nodes in 1st Hidden Layer	Nodes in 2nd Hidden Layer	Mean Square Error (MSE) [×10^{-4}]			
		log-log(1)	tan-tan(2)	log-tan(3)	tan-log(4)
3	3	141.30	117.55	6.7414	7.2229
	5	5.4388	6.8298	7.0791	6.4274
	7	6.6859	5.5082	5.5983	6.6292
	9	6.3631	6.5603	6.2066	6.6250
5	3	6.6076	6.9436	7.0079	5.6069
	5	6.5512	6.6080	5.2312	6.1628
	7	6.7937	6.3604	6.6079	6.8571
	9	6.0871	5.4009	6.6401	6.8571
7	3	6.5780	5.4294	42.574	26.935
	5	36.353	7.2599	6.6619	5.3287
	7	32.706	5.1269	1.1392	5.5199
	9	6.5507	6.4861	6.3260	6.7798
9	3	4.9698	5.3339	5.9890	6.7233
	5	6.2447	5.6917	6.5645	6.1474
	7	6.5773	4.4371	6.3697	6.5357
	9	4.8290	5.0833	6.5795	5.0078

(1) log-sigmoid transfer function in 1st and 2nd hidden layer

(2) tan-sigmoid transfer function in 1st and 2nd hidden layer

(3) log-sigmoid and tan-sigmoid transfer function in 1st and 2nd hidden layer, respectively

(4) tan-sigmoid and log-sigmoid transfer function in 1st and 2nd hidden layer, respectively

Table 4.4 MSE of various network structures: 3 layers

Nodes in 1st Hidden Layer	Nodes in 2nd Hidden Layer	Nodes in 3rd Hidden Layer	Mean Square Error (MSE) [$\times 10^{-4}$]
3	3	3	293.646
		5	5.8851
		7	6.3296
		9	26.6982
	5	3	6.2416
		5	248.198
		7	6.3224
		9	6.9727
	7	3	6.4698
		5	5.3558
		7	6.6531
		9	6.7970
	9	3	6.5793
		5	6.3281
		7	5.6535
		9	6.4608

Table 4.5 The summation of the optimal network structure

Network Object	Details
Training Algorithm	Back Propagation Algorithm
Network Structure	5-3-7-5-3
Transfer Function of 1st Hidden Layers	log-sigmoid
Transfer Function of 2nd Hidden Layers	log-sigmoid
Transfer Function of 3rd Hidden Layers	log-sigmoid
Transfer Function of Output Layer	linear
Input Data	T(t), T(t-1), T(t-2), C(t), C(t-1)
Output Data	C(t+1), μ_3^n(t+1), μ_3^s(t+1)

4.2.5 Network Validation

One of the problems that occur during neural network training is called overfitting. The error on the training set is driven to a very small value, but when new data is presented to the network the error is large. The network has memorized the training examples, but it has not learned to generalize to new situations. The validation vectors are used to stop training early if further training on the primary vectors will hurt generalization to the validation vectors. The error on the validation set is monitored during the training process. The validation error will normally decrease during the initial phase of training, as does the training set error. However, when the network begins to overfit the data, the error on the validation set will typically begin to rise. When the validation error increases for a specified number of iterations, the training is stopped, and the weights and biases at the minimum of the validation error are returned (Demuth et al., 1992-2006).

4.2.6 Simulating the network response to new inputs

Properly trained networks tend to give reasonable answers when presented with inputs that they have never seen. Typically, a new input leads to an output similar to the correct output for input vectors used in training that are similar to the new input being presented.

4.2.7 Summary of the Basic Procedure of Network Designing

There are many procedures of neural network designing; however, the basic steps of the artificial neural design are summarized in Figure 4.7.

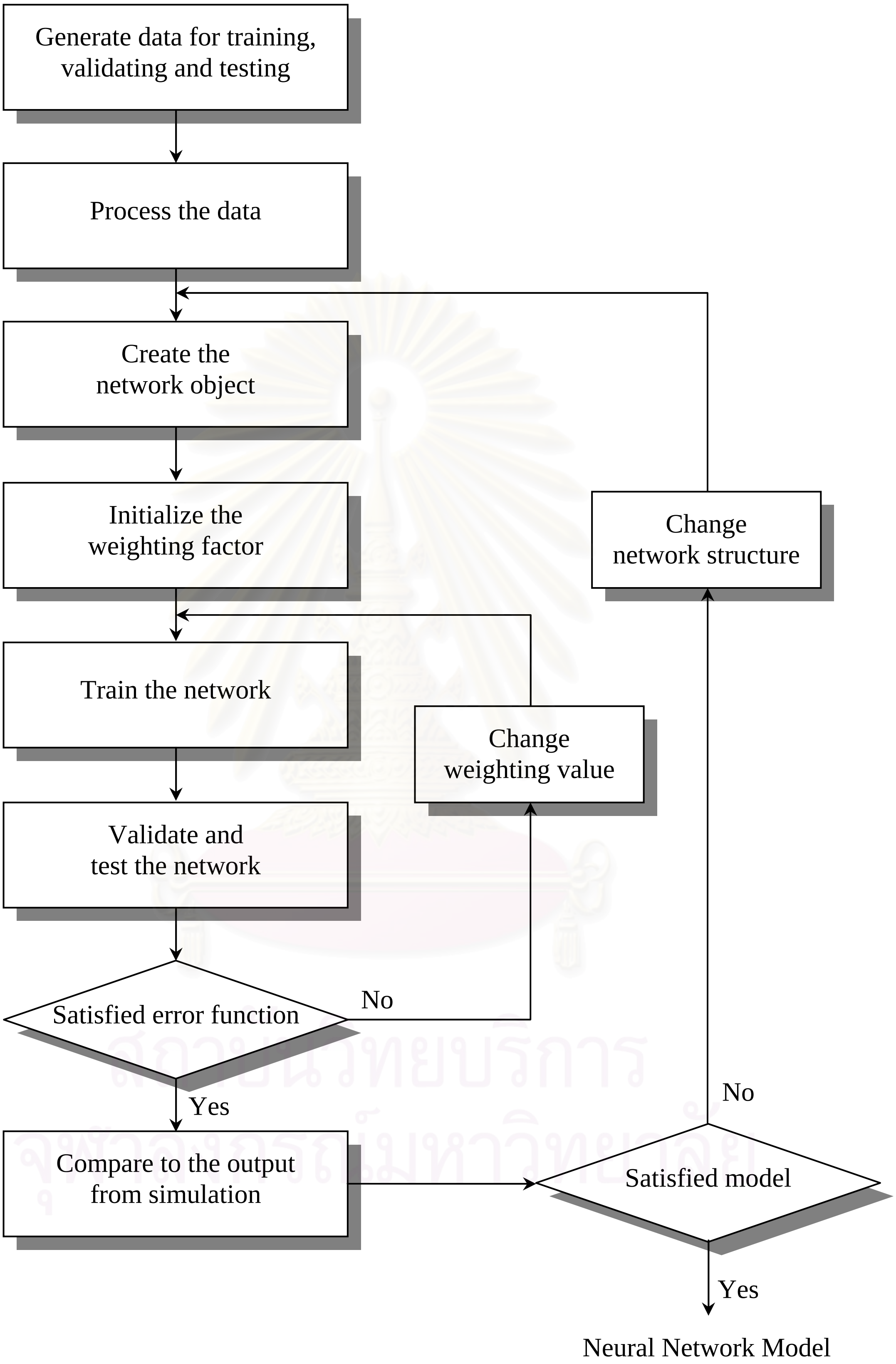

Generate data for training, validating and testing
Process the data
Create the network object
Initialize the weighting factor
Train the network
Change weighting value
Change network structure
Validate and test the network
Satisfied error function
No
Yes
Compare to the output from simulation
Satisfied model
No
Yes
Neural Network Model

Figure 4.7 Steps of neural network designing

4.3 Neural Network Predictor

It is known that some process variables in chemical industrial processes cannot be measured directly and difficult to predict. For this reason, a state predictor is proposed to deal with the problem. From the previous steps, the network was trained with the output data of future values of concentration, μ_3^n and μ_3^s. Therefore, the properly trained network will have an ability to predict the future values of the outputs for the presences of new inputs.

First of all, the new input information applied to the network has to be transformed. All subsequent inputs to the network need to be transformed using the same normalization with the previous training step that gets the optimum network. The mean and standard deviation of the training input are used to transform the new input as described in Equation 4.18. Denote that, $\text{Input}_{\text{norm}}$ is the normalized input, $\text{Input}_{\text{actual}}$ is the actual input, $\text{Mean}_{\text{input}}$ is mean of the training input, and $\text{Std}_{\text{input}}$ is the standard deviation of the training input.

$$\text{Input}_{\text{norm}} = \frac{\text{Input}_{\text{actual}} - \text{Mean}_{\text{input}}}{\text{Std}_{\text{input}}} \qquad (4.18)$$

After the network predicting, the predicted network output must also be denormalized to the original units by using mean and standard deviation of the training output. The denormalized algorithm is shown in Equation 4.19. Denote that, $\text{Output}_{\text{actual}}$ is the actual output, $\text{Output}_{\text{pred}}$ is predicted network output, $\text{Mean}_{\text{output}}$ is mean of the training output, and $\text{Std}_{\text{output}}$ is the standard deviation of the training output.

$$\text{Output}_{\text{actual}} = (\text{Output}_{\text{pred}} \times \text{Std}_{\text{output}}) + \text{Mean}_{\text{output}} \qquad (4.19)$$

In this work, the neural network predicted values are defined as a_j^5 which means the output value of the *j*th node of the fifth layer. Denote that the superscript and the subscript indicate the number of layer and the number of nodes, respectively. Signifying that, the first layer is an input layer, the second, third and forth layers are hidden layer and the fifth layer is an output layer. The calculation of a_j^5 is described in Equation 4.20.

$$a_j^5 \;=\; f^5\left(\sum_{k=1}^{5}(w_{jk}^5 a_k^4)+b_j^5\right) \qquad (4.20)$$

where f^5 is the transfer function of the fifth layer, w_{jk}^5 is the weighting factors of output from the *k*th node of the forth layer to the *j*th node of the fifth layer, and b_j^5 is the bias of the *j*th node in the fifth layer.

The outputs of the *j*th node of the third layer, a_j^4, is defined as

$$a_j^4 \;=\; f^4\left(\sum_{k=1}^{7}(w_{jk}^4 a_k^3)+b_j^4\right) \qquad (4.21)$$

a_j^3 is the outputs of the *j*th node of the third layer which is defined as

$$a_j^3 \;=\; f^3\left(\sum_{k=1}^{3}(w_{jk}^3 a_k^2)+b_j^3\right) \qquad (4.22)$$

a_j^2 is the outputs of the *j*th node of the second layer which is defined as

$$a_j^2 \;=\; f^2\left(\sum_{k=1}^{5}(w_{jk}^2 a_k^1)+b_j^2\right) \qquad (4.23)$$

From Equation 4.21 to 4.23, f^{i} is the transfer function of the *i*th layer, w^{i}_{jk} is the weighting factor of output from the *k*th node in the *(i-1)*th layer to the *j*th node in the *i*th layer, and b^{i}_{k} is the bias of the *k*th node in the *i*th layer. a^{1}_{j} in Equation 4.22 is indicated by the value of normalized neural network input. The transfer functions of each layer are presented in Table 4.6. The mean and standard deviation of each variable are shown in Table 4.7.

Table 4.6 The transfer functions of each layer in neural network predictor

Layer	Variables	Transfer Function	Equation
2nd Layer (Hidden Layer)	f^{2}	log-sigmoid	$f^{2}(x) = \frac{1}{1+e^{-x}}$
3rd Layer (Hidden Layer)	f^{3}	log-sigmoid	$f^{3}(x) = \frac{1}{1+e^{-x}}$
4th Layer (Hidden Layer)	f^{4}	log-sigmoid	$f^{4}(x) = \frac{1}{1+e^{-x}}$
5th Layer (Output Layer)	f^{5}	Linear	$f^{5}(x) = x$

Table 4.7 The mean and standard deviation of each variable

Variables		Normalized Parameters	
		Mean	Standard Deviation (Std.)
Input#1	T(t)	39.7057	6.0045
Input#2	T(t-1)	39.7051	6.0044
Input#3	T(t-2)	39.7044	6.0044
Input#4	C(t)	0.1533	0.0123
Input#5	C(t-1)	0.1533	0.0123
Output#1	C(t+1)	0.1533	0.0123
Output#2	μ_3^n(t+1)	4.9502×10^8	4.8931×10^8
Output#3	μ_3^s(t+1)	6.1336×10^9	2.6095×10^9

The weighting factors and biases of the optimum neural network architecture are used to design the neural network predictor. The inputs are normalized and calculated through the previous steps. Then, the network predictor will predict the output. The weighting factors and biases of each layer in each node are presented below in Table 4.8 to Table 4.12.

After obtaining the optimal structure of the neural network, it was test with another input for comparing the network output values to the process values. In this case the testing data presented is the linear cooling temperature profile. It was found that the network can approximate the output data near to the process data. The testing comparison is shown in Figure 4.8.

Table 4.8 Weighting factors in the second layer of the optimum neural network

w_{jk}^2	k = 1	k = 2	k = 3	k = 4	k = 5
j = 1	0.0156	-0.0882	0.0702	127.7707	-87.3093
j = 2	-0.1505	0.1271	0.1791	-72.8489	79.1908
j = 3	0.1131	-0.1900	-0.0898	-84.7598	-85.1862

Table 4.9 Weighting factors in the third layer of the optimum neural network

w_{jk}^3	k = 1	k = 2	k = 3
j = 1	-4.1961	4.7436	8.6398
j = 2	7.4927	2.4918	-7.2393
j = 3	-5.4442	-6.9761	6.0374
j = 4	-0.0403	-10.2622	3.0726
j = 5	-4.2198	-9.7713	1.2130
j = 6	4.9420	-6.0592	-7.3225
j = 7	7.6273	5.2748	5.3626

Table 4.10 Weighting factors in the forth layer of the optimum neural network

w_{jk}^{4}	k = 1	k = 2	k = 3	k = 4	k = 5	k = 6	k = 7
j = 1	-2.3156	1.0262	0.3873	-4.7636	1.7478	0.1589	-4.1631
j = 2	2.7648	1.7387	1.3451	-2.0667	-4.6402	-0.4556	-3.3439
j = 3	1.6268	-3.2267	1.8943	2.8439	-4.3751	2.0707	-1.2269
j = 4	-2.9584	1.4128	1.8349	-1.9938	-5.0497	0.8534	-2.3131
j = 5	0.9361	-3.0222	3.4543	3.9113	-2.5011	0.0844	2.4831

Table 4.11 Weighting factors in the fifth layer of the optimum neural network

w_{jk}^{5}	k = 1	k = 2	k = 3	k = 4	k = 5
j = 1	-0.3721	0.0058	0.8351	0.8165	0.5253
j = 2	0.2764	0.8954	-0.7738	-0.6872	0.4436
j = 3	0.9731	0.6561	0.6243	-0.7558	0.3033

Table 4.12 Biases of the optimum neural network

b_j^i	i = 2	i = 3	i = 4	i = 5
j = 1	-9.5642	0.7625	7.4849	0.5080
j = 2	-7.1820	-4.9434	0.5671	0.3263
j = 3	36.1326	4.9769	0.1964	0.7670
j = 4		3.6150	2.3451	
j = 5		4.6037	0.8509	
j = 6		7.7906		
j = 7		-3.7761		

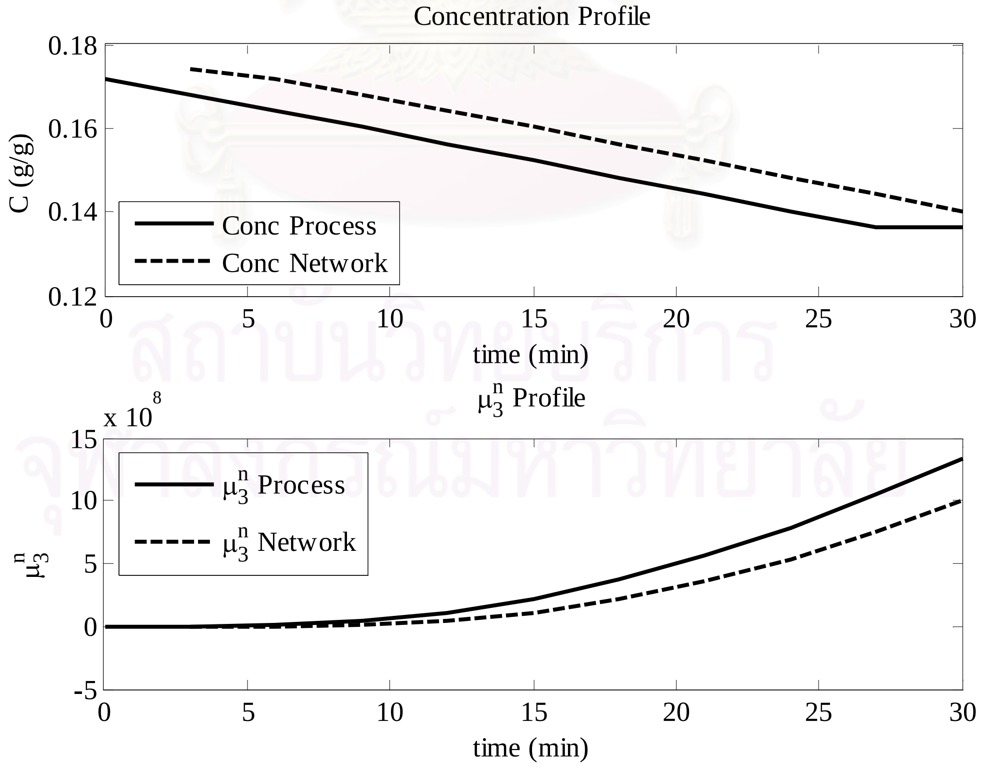


Figure 4.8 The neural network testing comparison

CHAPTER V

NEURAL NETWORK-BASED OPTIMAL CONTROL OF BATCH CRYSTALLIZATION

In this chapter, the neural network model will be applied with the optimal control to deal with the limitation of the optimal control. The optimal control of a batch crystallizer of potassium sulfate production has been studied in this work. The solution of the optimal control problem is computed using a sequential model solution and optimization method.

The first step of using neural network in optimal control for predicting is to train a neural network for representing the forward dynamics of the plant. The prediction error between the plant output and the neural network output is used as the neural network training signal. The neural network predictor used previous inputs and previous plant outputs to predict future values of the plant output. The mathematical models in section 4.1 are used to generate data for training the neural network. After the properly training, the network will have an ability to describe the process. The obtained network will be applied within the optimal control for the purpose of prediction. The basic algorithm the optimal control with neural network predictor is summarized as the followings.

Step1: Problem formulation to specify the objective function and the initial states of the system.

Step2: Calculation of the control profile that optimizing the objective function over the batch stage.

Step3: Implement of the obtained optimal controls to process.

5.1 Problem Formulation

In the industrial crystallizers, the fine particles usually cause difficulties in downstream processing equipment (e.g., filtration) and affect both product quality and process economics. Excessive fines may also require a relatively long batch run time to achieve the desired final size of the product (Shi et. al, 2006). For this reason, this section will focus on a developing of an optimal control system based on neural network predictor to control the particle size distribution of the final product crystals.

The control objective is to determine the optimal temperature profile that minimize the total volume of nucleated fine particles i.e., the third moment of the crystals formed by nucleation (μ_3^n) and maximize the total volume of the crystals growing from the seeds (μ_3^s) at the end of the operation. Manipulated input limitation and concentration specifications are incorporated as input and state constraints on the optimization problem, which takes the form as the followings.

Problem:

$$Max\ \frac{\mu_3^s(t_f)}{\mu_3^n(t_f)} \tag{5.1}$$

Subject to

$$T_{\min} \le T \le T_{\max} \tag{5.2}$$

$$C_S \le C \le C_m \tag{5.3}$$

$$\mathrm{C(t=0)} = 0.1742 \tag{5.4}$$

$$\mathrm{T(t=0)} = 50\ {}^{o}\mathrm{C} \tag{5.5}$$

$$\left\|\frac{dT}{dt}\right\| \leq k_5 \tag{5.6}$$

$$\mu_3^s(t_f) \geq V_1 \tag{5.7}$$

where μ_3^n is the total volume of nucleated fine particles, C is the solution concentration achieved from the neural network predictor that is developed in section 4.3. C_S and C_m are defined by the Equation 4.6 and Equation 4.7 in section 4.1. The constant k_5 which is the maximum gradient of the reactor temperature is chosen to be 2 ^oC/min. V_1, chosen as 8.3301 × 10^9 in the simulations, denotes the lower bound on the total volume of the crystals growing from the seeds. Such constraint on μ_3^s(t_f) represents a desirable quality of the final product. In the simulation, Δt is chosen as 30 s. The final batch time (t_f) is 30 minutes.

5.2 Calculation of Optimal Control Profile

The neural network model predicts the process response over a specified time horizon. The predictions are used by a numerical optimization program to determine the control signal that minimizes the total volume of nucleated fine particles, μ_3^n, as described in section 5.1. The control structure of the optimal control with neural network predictor for a batch crystallizer is shown in Figure 5.1. The artificial neural network is used as a state predictor to predict the future values of the solution concentration, the third moment of the crystals formed by nucleation (μ_3^n) and total volume of the crystals growing from the seeds (μ_3^s). The controller consists of the neural network plant model and the optimization block. The optimization block determines the temperature profile that maximize μ_3^s / μ_3^n at the final time and then the optimal profile will be input to the process.

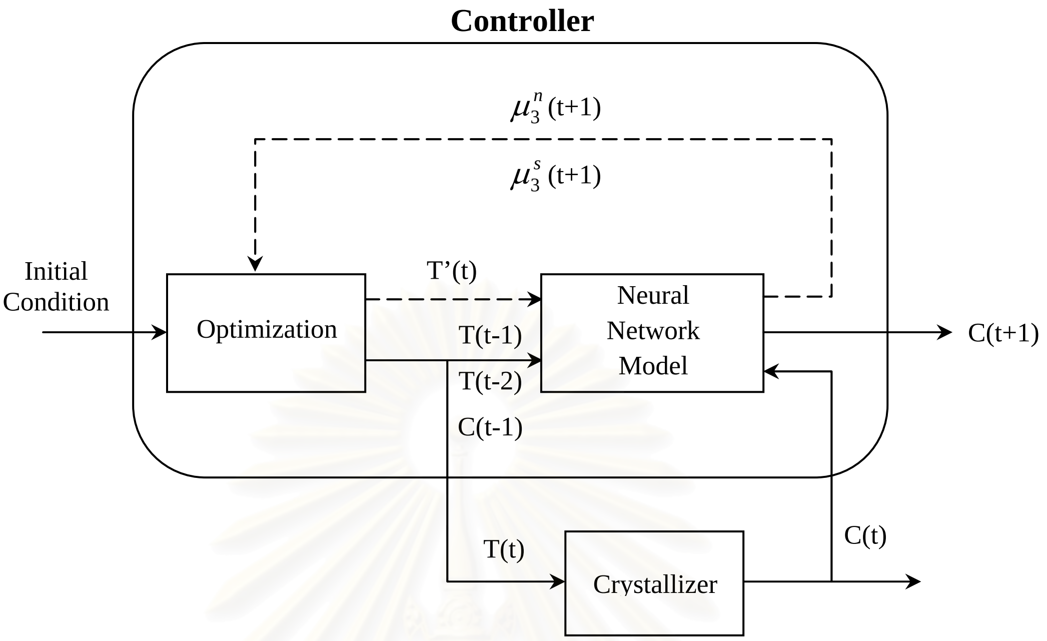


Figure 5.1 Schematic diagram of the control process

A further difficulty in batch process control is that product quality variables usually cannot be measured on-line and can only be obtained through laboratory analysis after a batch has finished (Zhang, 2005). For this reason, the recursive operation of neural network should be applied to the process output predictor that in this work. The control structure of the optimal control with recursive neural network predictor for a batch crystallizer is shown in Figure 5.2.

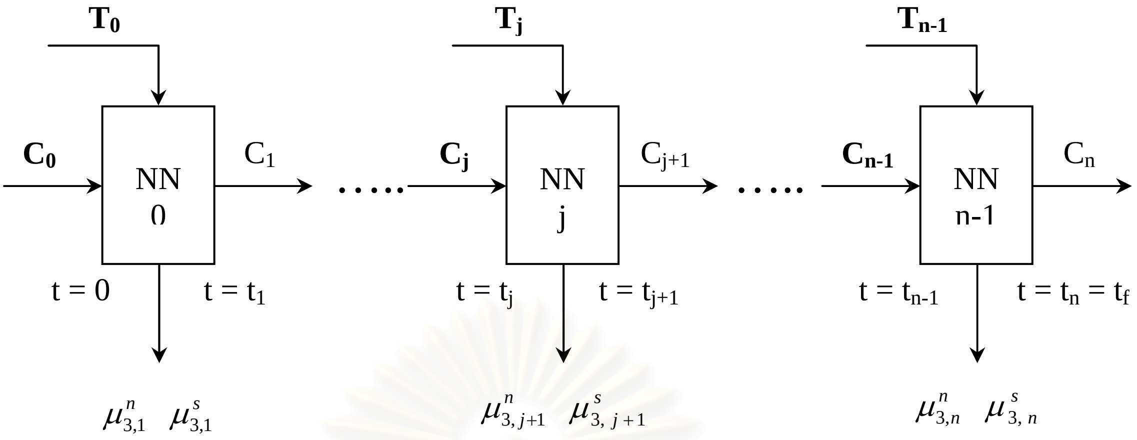

T0
Tj
Tn-1
C0
NN 0
C1
Cj
NN j
Cj+1
Cn-1
NN n-1
Cn
t = 0
t = t1
t = tj
t = tj+1
t = tn-1
t = tn = tf
$\mu_{3,1}^{n}$ $\mu_{3,1}^{s}$
$\mu_{3,j+1}^{n}$ $\mu_{3,j+1}^{s}$
$\mu_{3,n}^{n}$ $\mu_{3,n}^{s}$

Figure 5.2 The structure of recursive neural network predictor

Figure 5.2 shows the recursively arrangement of n approaches of neural network predictor. This operation leads to the continuous in the time domain. The predicted state variables are solution concentration (C), the third moment of the crystals formed by nucleation (μ_3^n) and total volume of the crystals growing from the seeds (μ_3^s). Let C_j, $\mu_{3,j}^n$, $\mu_{3,j}^s$ represent the status of the crystallization process at time t = t_j. The neural network output at time t_{j+1} (C_{j+1}, $\mu_{3,j+1}^n$ and $\mu_{3,j}^s$) depend on the input at time t = t_j which is the matrix of solution concentration that consists of solution concentration at time t_j and t_{j-1}. The sampling time interval, $\Delta t_j = t_j - t_{j-1}$, is assumed to be constant. The series of the network imply that the output concentration of *(j-1)*th network is used to be an input for the *j*th network. This procedure is repeated for the next sampling time until the end of the operation. The matrix of reactor temperature is an external input to the *j*th process, consists of reactor temperature at time *j*th, *(j-1)*th and *(j-2)*th. The nonlinear relationship between the input and output of *j*th process can be described by neural network model. The same model can be used to describe the process for all n intervals, if the model is trained by using the pair of input and output for all j = 0, 1, 2,…, n (Chaudhuri and Modak). Thus, same neural network model can be used recursively to describe crystallization process during the entire operation (0, t_f) once the initial conditions are specified.

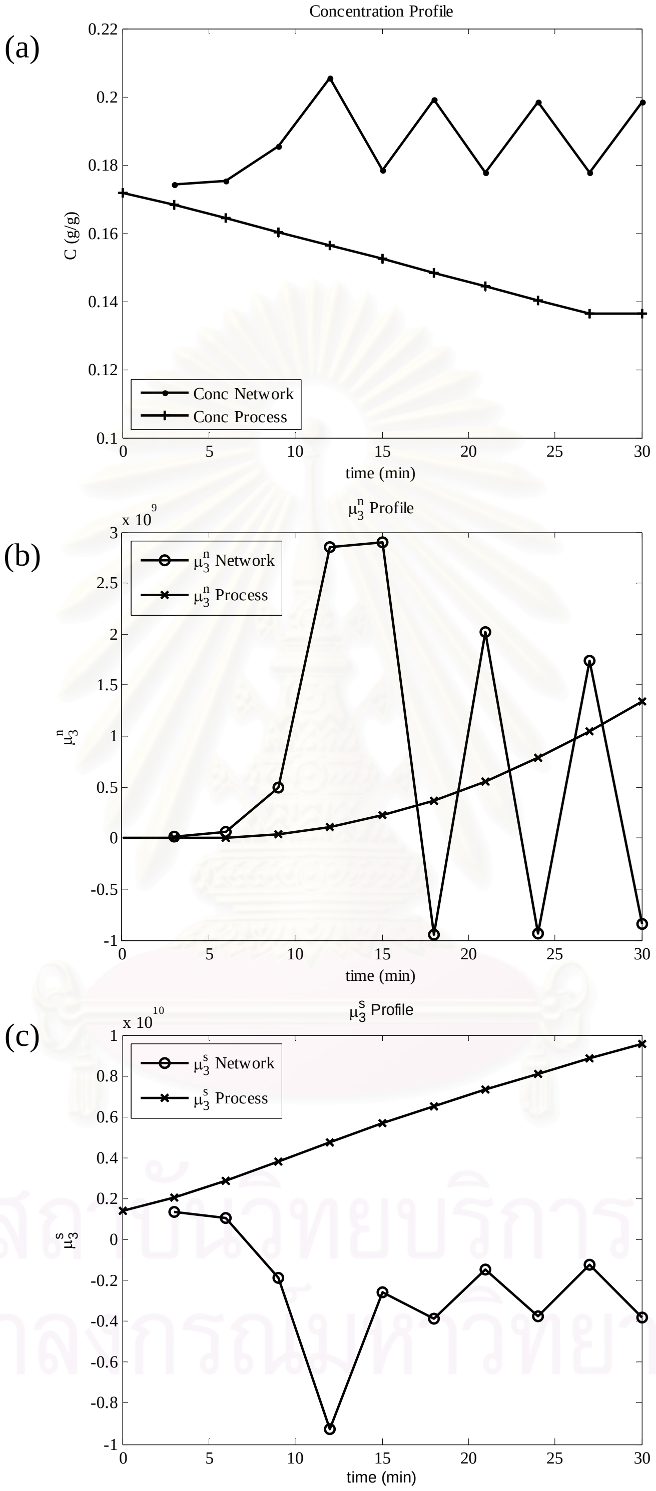


Figure 5.3 Results of 1 layer simulation applying with recursive neural network (a) Concentration profile (b) μ_3^n profile (c) μ_3^s profile

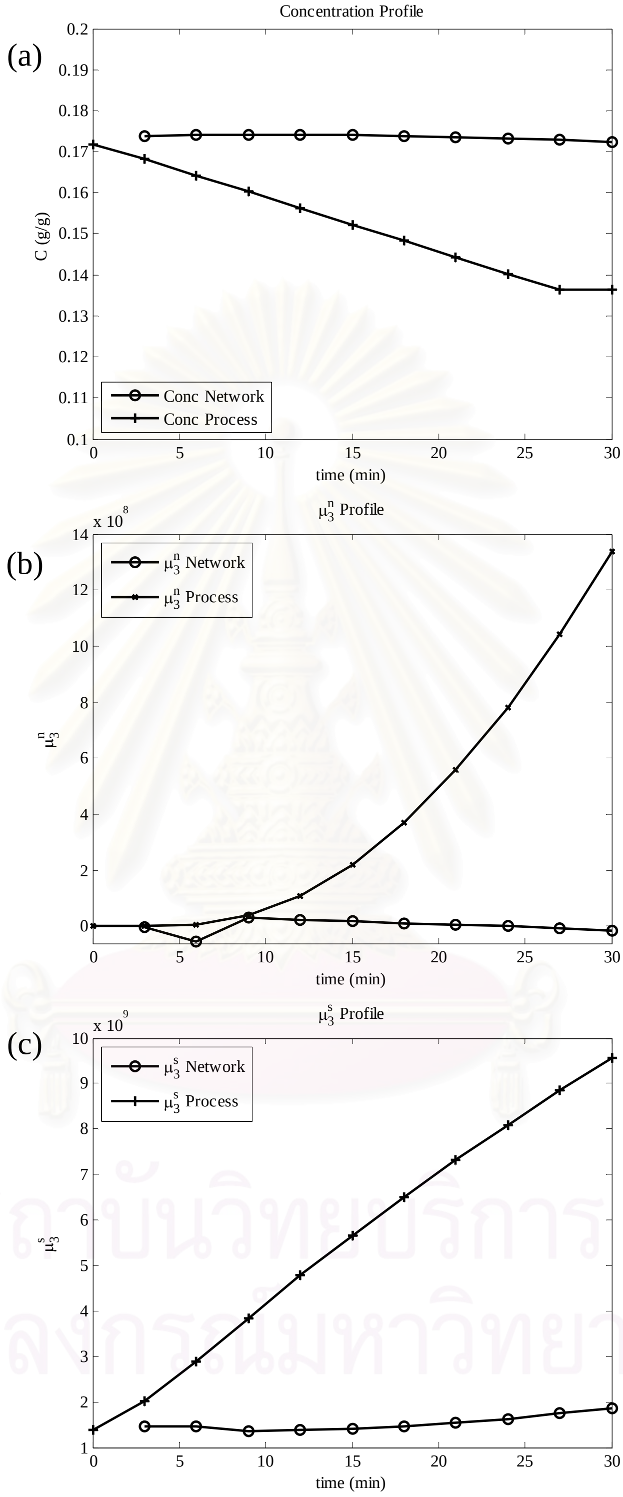


Figure 5.4 Results of 2 layer simulation applying with recursive neural network (a) Concentration profile (b) μ_3^n profile (c) μ_3^s profile

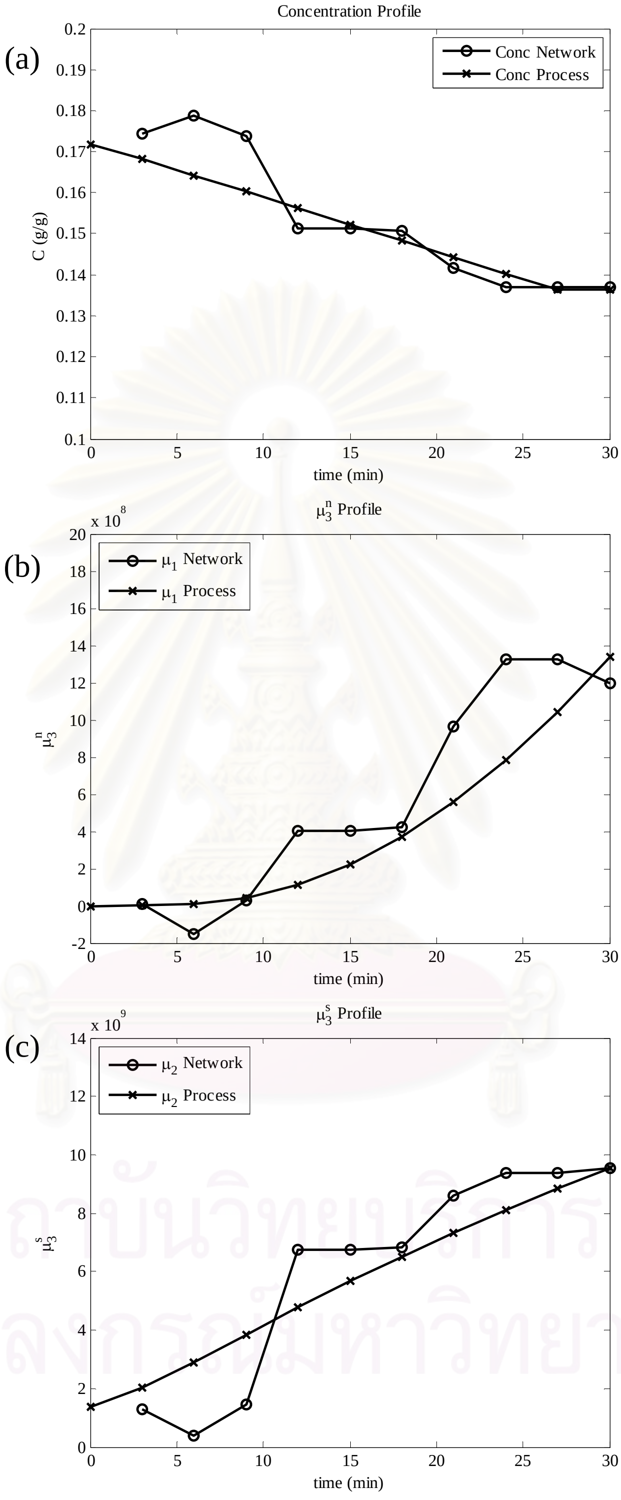


Figure 5.5 Results of 3 layer simulation applying with recursive neural network (a) Concentration profile (b) μ_3^n profile (c) μ_3^s profile

Figures 5.3 – 5.5 show the simulation results of network outputs profile when applying with recursive neural network. These simulation results show that 3 layer neural network structures give closely the predicting profile but 1 layer network and 2 layer network cannot give the appropriate predicting.

5.3 Implementation of optimal control

The calculation of the optimal temperature profile gives the minimized μ_3^n at the final batch time. The optimal temperature profile with 10 time interval is shown in Figure 5.6.

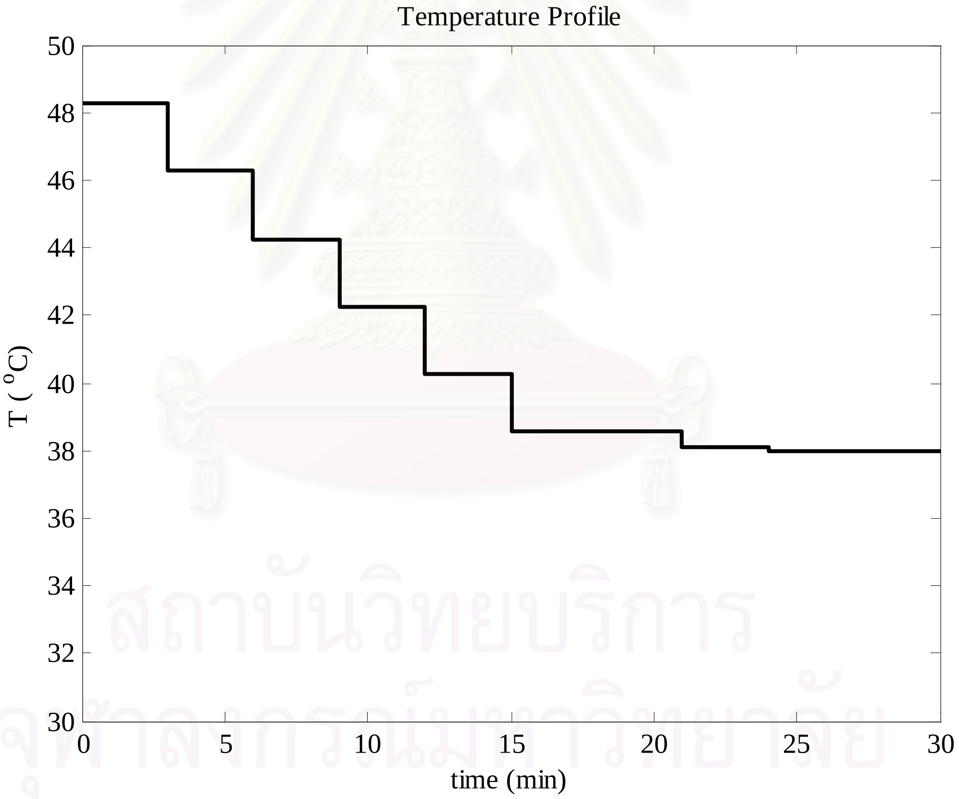


Figure 5.6 Optimal temperature profile

The simulation results of the optimal temperature profile with 10 intervals compare to the results of linear cooling is presented in Figure 5.7 - 5.9. Figure 5.7 shows the evolution of the solution concentration and the metastable boundary of the operations. Signifying that, the constraints on the solution concentration is respected during the evolution of the optimal control profiles. In Figure 5.8 and 5.9, the comparison of μ_3^n and μ_3^s in case of 2 different control strategy are presented. In Figure 5.8, the dotted line and solid line represent the value of μ_3^n in case of applied with linear cooling profile and 10 intervals optimal control profile, respectively. Note that, the optimal temperature profile give the lower value of μ_3^n compared to the linear cooling profile.

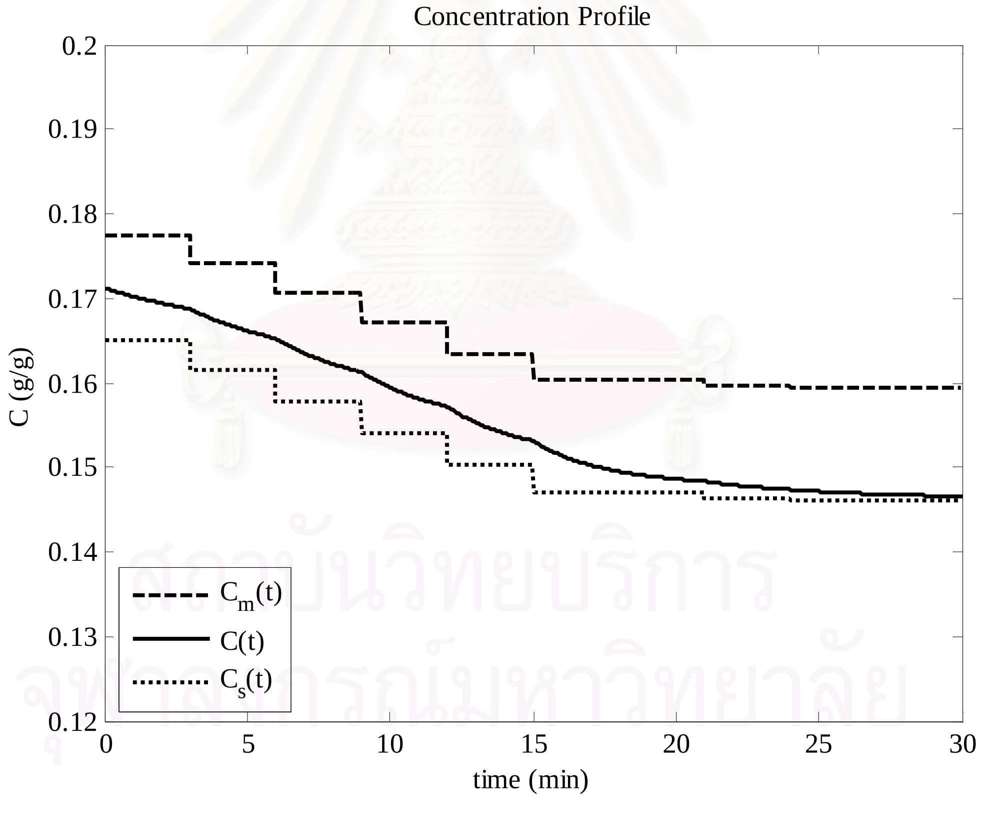


Figure 5.7 Evolution of concentration by applying optimal control strategies

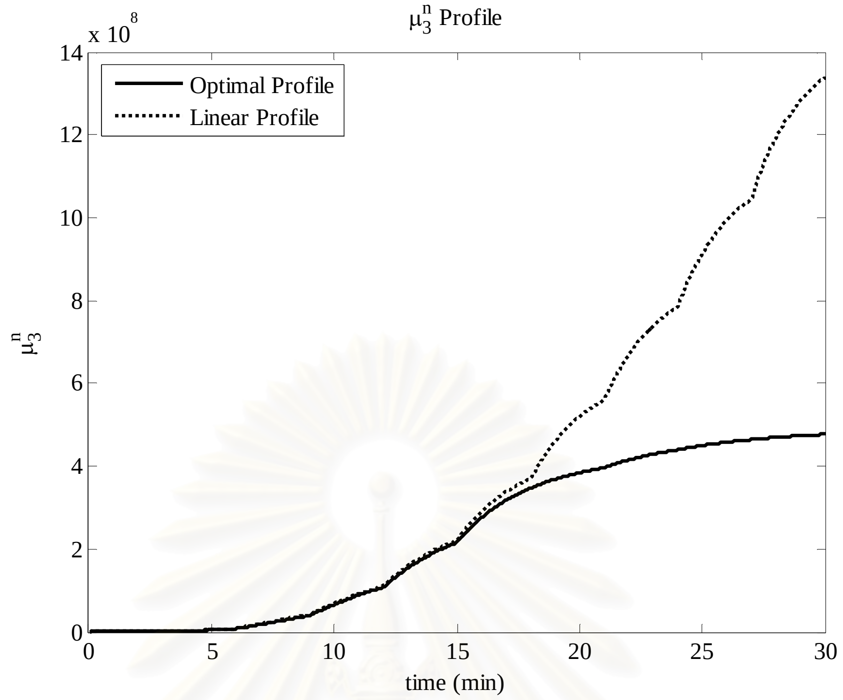


Figure 5.8 Evolution of μ_3^n by applying 2 different control strategies

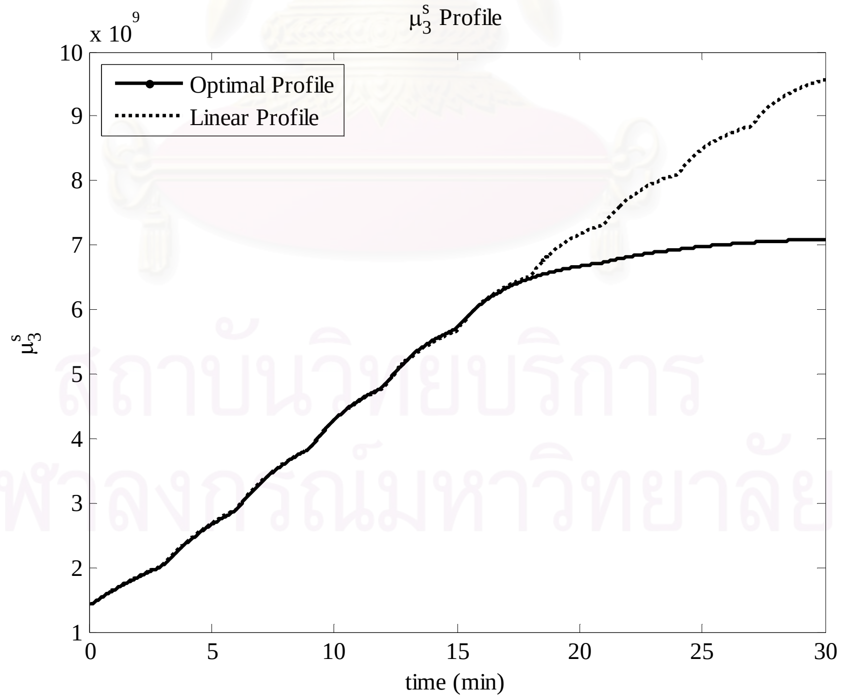


Figure 5.9 Evolution of μ_3^s by applying 2 different control strategies

Table 5.1 lists the value of μ_3^n and μ_3^s obtained under three different control strategies: (1) linear cooling control, (2) optimal control with an objective to maximize $\mu_3^s(t_f)/\mu_3^n(t_f)$, (3) optimal control with an objective to maximize $\mu_3^s(t_f)/\mu_3^n(t_f)$ based on mathematical model (Shi et. al, 2006), respectively. Comparing the results of these control strategies, it is clear that the optimal control strategy can yield the lower volume of fines (nucleation formed crystals) compare to the linear cooling strategy, while the crystals growing from the seeds in final product still satisfy the product quality requirement. This optimal control strategy is shown to be able to reduce the total volume of the fines by 46.53% compared to a linear cooling strategy and 73.25% compare to the control that based on mathematical model.

Table 5.1 Comparison between the simulation results of various control strategies

Control Strategy	$\mu_3^n(t_f)$	$\mu_3^s(t_f)$
Linear cooling control	8.917×10^9	9.1121×10^9
Optimal control with an objective to maximize $\mu_3^s(t_f)/\mu_3^n(t_f)$ based on neural network	4.7681×10^8	9.6841×10^9
Optimal control with an objective to maximize $\mu_3^s(t_f)/\mu_3^n(t_f)$ based on mathematical model	1.7828×10^9	1.0545×10^{10}

CHAPTER VI

CONCLUSION AND RECOMMENDATIONS

6.1 Conclusion

In this research, the control of the potassium sulfate production process in a batch crystallizer has been studied. The optimal control strategy is applied to obtain the minimum value of total volume of nucleated fine particles i.e., the third moment of the crystals formed by nucleation (μ_3^n) and maximize the total volume of the crystals growing from the seeds (μ_3^s) at the end of the operation. Due to the highly nonlinear and complex dynamic behavior, a linear controller may not give the optimal performance when applied to the actual process. A further difficulty in batch process control is that product quality variables usually cannot be measured on-line and can only be obtained through laboratory analysis after a batch has finished. To realize this fact, the optimal control with recursive neural network predictor is proposed to modify the reactor temperature profile for improving the control performance.

Recursive artificial neural network is applied to predict the solution concentration and the third moment of the crystals formed by nucleation (μ_3^n). The predicted concentration from *(j-1)*th process is used to be an input for the *j*th process. Then the same neural network model can be used recursively to describe crystallization process during the entire operation. A multilayer feedforward neural network is trained by Levenberg-Marquardt Backpropagation algorithm. The optimum topology of an artificial neural network (5-3-7-5-3) is employed as a neural network predictor.

In the optimal control strategy, the solution of the optimal control problem is computed using a sequential model solution and optimization method. From the simulation results, it can be seen the recursive neural network-based optimal control provide a better control performance compared to the linear cooling control or randomly decreasing temperature control.

6.2 Recommendation

In this work, recursive neural network predictor is only used as an off-line optimal controller. For the future work, the proposed on-line optimal control with recursive neural network predictor should be applied to improve the performance of the controller for the control of many complicated system such as crystallization process, complex fermentation or polymerization process.

REFERENCES

Åkesson, B. M. and Toivonen, H. T. A neural network model predictive controller. Journal of Process Control (2006).

Ashobi, M. Modeling and Control of a Continuous Crystallization Process Using Neural Networks and Model Predictive Control. A Thesis Submitted to the College of Graduate Studies and Research in Partial Fulfillment of the Requirements For the Degree of Doctor of Philosophy in the Department of Electrical Engineering University of Saskatchewan. 1995.

Banga, J. R. and Carrasco, E. F. Dynamic optimization of batch reactors using adaptive stochastic algorithms. Ind. Eng. Chem. Res. 36 (1997): 2252-2261.

Braatz, R. D. Advanced Control of Crystallization Process. Annual Reviews in Control. 26 (2002): 87-99.

Chaudhuri, B. and Modak, J. M. Optimization of fed-batch bioreactor using neural network model. Department of chemical Engineering, Indian Institute of Science, Bangalore, India. 2006.

Costa, C. B. B., Da Costa, A. C. and Filho, R. M. Mathematical modeling and optimal control strategy development for an adipic acid crystallization process. Chemical Engineering and Processing. 44 (2005): 737–753.

Demuth, H., Beale, M. and Hagan, M. Neural Network Toolbox For Use with MATLAB. Neural Network Toolbox User's Guide. Copyright 1992—2006 by the MathWorks, Inc.

Eek, R.A., Both, J. A. and Van den Hof, P. M. J. Closed-loop identification of a continuous crystallization process. Mechanical Engineering Systems and Control Group, Delft University of Technology, Netherlands. 1994.

Fujiwara, M., Nagy, Z. K., Chew, J. W. and Braatz, R. D. First-principles and direct design approaches for the control of pharmaceutical crystallization, Journal of Process Control. 15 (2005): 493-504.

Georgieva, P. and De Azevedo, S. F. Application of Feed Forward Neural Networks in Modeling and Control of a Fed-Batch Crystallization Process. Transactions on Engineering, Computing and Technology. V12 March 2006: ISSN 1305-5313.

Georgieva, P. and De Azevedo, S. F. Neural Network-Based Control Strategies Applied to a Fed-Batch Crystallization Process. International journal of computational intelligence. Vol. 3 No. 3 (2006) ISSN: 1304-2386.

Hagan, M. T., Demuth, H. B. and De Jesús, O. An Introduction to the Use of Neural Networks in Control Systems, School of Electrical & Computer Engineering. 2002.

Hu, Q., Rohani, S., Wang, D. X. and Jutan, A. Optimal control of a batch cooling seeded crystallizer. Powder Technology. 156 (2005): 170 – 176.

Jones, A. G. and Mullin, J. W. Programmed cooling crystallization of potassium sulfate solutions. Chemical Engineering Science. 29 (1974): 105.

Kraft, M. Modeling of Particulate Processes. Department of Chemical Engineering University of Cambridge. KONA No.23 (2005). 18-35.

McCabe, W. L., Smith, J. C. and Harriott, P. Unit Operation of Engineering, 6th edition. McGraw-Hill. 2001.

Miller, S. M. and Rawlings, J. B. Model identification and control strategies for batch cooling crystallizers. A.I.Ch.E. Journal. 40 (1994): 1312–1327.

Mjalli, F. S., Al-Asheh, S., Alfadala, H. E. Use of artificial neural network black-box modeling for the prediction of wastewater treatment plants performance. Journal of Environmental Management. 83 (2007): 329–338.

Nagy, Z. K. Model based control of a yeast fermentation bioreactor using optimally designed artificial neural networks. Chemical Engineering Journal. 127 (2007): 95–109.

Nowee, S. M., Abbas, A. and Romagnoli, J. A. Optimization in seeded cooling crystallization: A parameter estimation and dynamic optimization study. Chemical Engineering and Processing. 46 (2007): 1096–1106.

Shi, D., El-Farra, N. H., Li, M., Mhaskar, P. and Christofides, P. D. Predictive Control of Particle Size Distribution in Particulate Processes. Chemical Engineering Science. 61 (2006): 268 – 281.

Shi, D., Mhaskar, P., El-Farra, N. H. and Christofides, P. D. Predictive Control of Particle Size Distribution in Protein Crystallization. American Control Conference. June 8-10 2005.

Shomchoam, N. On-line Optimal Control of Ethanol Production in a Fed-batch Reactor by Using Neural Network Estimator. Department of Chemical Engineering, Chulalongkorn University, Bangkok, Thailand. 2006.

Wu, W., Chang, J. X., Wu, C. J., Hsu, W. C. Reduced Neural Model Predictive Control Strategies for a Class of Chemical Reactors. Journal of Chemical Engineering of Japan. Vol. 40, No. 5 (2007): 422-431.

Zhang, G. P. and Rohani, S. On-line optimal control of a seeded batch cooling crystallizer. Chemical Engineering Science. 58 (2003): 1887 – 1896.

Zhang, J. Batch Process Modeling and Optimal Control Based on Neural Network Models. Acta Automatica Sinica. Vol. 31, No. 1. January 2005.

APPENDICES

APPENDIX A

BACKPROPAGATION LEARNING ALGORITHM

Backpropagation is the most widely used learning algorithm in an artificial neural network. In this algorithm, the error between neural network predicted output and the actual target is propagated backward from the output layer to the hidden layers and finally to the input layer. The weights and biases are changed in the direction of minimizing the prediction error.

For the multilayer feed-forward neural networks, the output of the first layer becomes the input of the following layer. The equations that describe of this operation are showed as the follows.

$$a^{m+1} = f^{m+1}(W^{m+1} a^{m} + b^{m+1}) \qquad \text{for } m = 0, 1, \ldots, M-1 \qquad \text{(A.1)}$$

where M is the number of layers of the network. The neurons in the first layer receive external inputs:

$$a^{0} = p \qquad \text{(A.2)}$$

The outputs of the neurons in the last layer of the network are considered as the network outputs:

$$a = a^{M} \qquad \text{(A.3)}$$

A.1 Performance Index

The backpropagation algorithm for multilayer feed-forward neural networks use the mean square error as a criterion which is shown in Eq. (A.4)

$$\begin{aligned} F(x) = E[e^2] &= E[(t-a)^2] \\ &= E[(t-a)^T(t-a)] \qquad \text{(A.4)} \end{aligned}$$

where *x* is the vector of network weights and biases, *t* and *a* are the corresponding target output and actual output respectively. The steepest descent algorithm for the approximate mean square error is shown as the follows:

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha \frac{\partial \overline{F}}{\partial w_{i,j}^m} \qquad \text{(A.5)}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha \frac{\partial \overline{F}}{\partial w_i^m} \qquad \text{(A.6)}$$

where α is the learning rate.

From Equation (A.5) and Equation (A.6), the error is an indirect function of the weights in the hidden layers. Therefore, the chain rule is used to determine the error gradient as the following

$$\frac{\partial \overline{F}}{\partial w_{i,j}^m} = \frac{\partial \overline{F}}{\partial n_i^m} \times \frac{\partial n_i^m}{\partial w_{i,j}^m} \qquad \text{(A.7)}$$

$$\frac{\partial \overline{F}}{\partial b_i^m} = \frac{\partial \overline{F}}{\partial n_i^m} \times \frac{\partial n_i^m}{\partial b_i^m} \qquad \text{(A.8)}$$

The second term in each of these equations can be easily computed, since the net input to layer *m* is an explicit function of the weights and biases in that layer:

$$n_i^m = \sum_{j=1}^{S^{m+1}} w_{i,j}^m a_j^{m-1} + b_i^m \qquad \text{(A.9)}$$

Therefore,

$$\frac{\partial n_i^m}{\partial w_{i,j}^m} = a_j^{m-1}, \quad \frac{\partial n_i^m}{\partial b_i^m} = 1 \tag{A.10}$$

Define

$$S_i^m = \frac{\partial \overline{F}}{\partial n_i^m} \tag{A.11}$$

which is the sensitivity of $\overline{F}$ to changes in the *i*th element of the net input at layer *m*, then Equation (A.7) and Equation (A.8) can be simplified to

$$\frac{\partial \overline{F}}{\partial w_{i,j}^m} = s_i^m a_j^{m-1} \tag{A.12}$$

$$\frac{\partial \overline{F}}{\partial b_i^m} = s_i^m \tag{A.13}$$

Therefore, the approximate steepest descent algorithm is expressed as the follows.

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha\, s_i^m a_j^{m-1} \tag{A.14}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha\, s_i^m \tag{A.14}$$

APPENDIX B

LEVENBERG - MARQUARDT TRAINING ALGORITHM

The Levenberg-Marquardt method was designed to approach second order training speed without having the computing of the Hessian matrix. When the performance function has the form of a sum of squares which is typical in training feedforward networks, then the Hessian matrix can be approximated as

$$H = J' J \qquad \text{(A.15)}$$

and the gradient can be computed as:

$$\nabla f = J' e \qquad \text{(A.16)}$$

where J is the Jacobian matrix, which contain the first derivatives of the network errors with respect to the weights and biases, and e is a vector of network errors. The Jacobian matrix can be computed through a standard backpropagation technique that is much less complex than the computing of the Hessian matrix.

The Levenberg-Marquardt method uses this approximation to the Hessian matrix in the following Newton like update.

$$x^{k+1} = x^{k} - [J' J + \mu I]^{-1} J' e \qquad \text{(A.17)}$$

When the scalar μ is zero, this is just a Newton’s method using the approximate Hessian matrix. When μ is large, this becomes gradient descent with a small step size. Newton’s method is faster and more accurate near an error minimum, so the aim is to shift towards Newton’s method as quickly as possible. Thus, μ is decreased after each successful step (reduction in performance function) and is increased only when a tentative step would increase the performance function. In this way, the performance function will always be reduced at each of iteration in the algorithm.

APPENDIX C

LIST OF PUBLICATIONS

National conference

1. Linda Thanasinthana and Amornchai Arpornwichanop. Neural Network-Based Model for Predicting Product Properties of a Batch Crystallization Process. The 17th Thailand Chemical Engineering and Applied Chemistry Conference, Chiangmai, Thailand, Oct. 29-30, 2007: PCF5_1-3 (IN THAI).

VITA

Miss Linda Thanasinthana was born in Bangkok, Thailand on December 21, 1983. She received the Bachelor Degree in Chemical Engineering from Chulalongkorn University in 2006. After that she continued to study in the Graduate School of Chulalongkorn University to pursue the Master Degree of Engineering in Chemical Engineering and completed in 2008.