



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

공학박사 학위논문

**Modifier adaptation schemes for
data-driven optimization of
chemical and biological processes
under model-plant mismatch**

모델 공정 불일치 상황에서 화학 생물 공정의
데이터 기반 최적화를 위한 개선형 적응법

2018년 2월

서울대학교 대학원
화학생물공학부
정 동 휘

Modifier adaptation schemes for data-driven optimization of chemical and biological processes under model-plant mismatch

지도교수 이 종 민

이 논문을 공학박사 학위논문으로 제출함

2017년 12월

서울대학교 대학원

화학생물공학부

정 동 휘

정 동 휘의 공학박사 학위논문을 인준함

2017년 12월

위 원 장 _____ (인)

부위원장 _____ (인)

위 원 _____ (인)

위 원 _____ (인)

위 원 _____ (인)

Abstract

Modifier adaptation schemes for data-driven optimization of chemical and biological processes under model-plant mismatch

Dong Hwi Jeong

School of Chemical and Biological Engineering

The Graduate School

Seoul National University

Most advanced processes in chemical and biological industry struggle to reduce the production cost while satisfying several constraint conditions. Operational optimization is a key to reducing production cost and satisfying requirements of quality and safety. Typical process optimization uses mathematical models and numerical procedures to compute an optimal solution. However, due to time and money constraints, it is rarely possible to create a model that perfectly matches the plant in the real industrial process. As a result, model-plant mismatch problem is an inevitable issue during the process optimization. The model-plant mismatch is important especially because it is closely related to the economic competitiveness of the product and the satisfaction of process constraints. Therefore, it is important to deal with model-plant mismatch both in terms of optimization and process safety.

To solve these problems caused by the model-plant mismatch, several methods have been proposed. First, there have been studies focusing on the improvement of the model itself. The modifications can be implemented in several ways such as addition of new coefficients or utilization of intracellular information. Second, two-step approach updating model parameters and performing optimization iteratively has been implemented to solve the optimality loss problem in the general industries. In this approach, the parameter-updated model is expected to yield a better description of the real plant near the current operating mode. However, it also has limitations such as (1) there should exist little structural or parametric model-plant mismatch, and (2) the changes in operating conditions should be large enough to provide sufficient excitations for parameter estimation.

In order to overcome these limitations of the conventional approaches, an iterative data-driven optimization method called modifier adaptation has been proposed. This method adjusts the model by adding the zeroth and first order error between model and plant and optimizes the process iteratively. It has been proven that the converged solution by the adjusted model satisfies the necessary conditions of optimality of the plant equation, even under model-plant mismatch. Due to this advantage, modifier adaptation has been actively studied and applied recently.

In this thesis, 3 issues of the modifier adaptation are discussed. First, a feedforward decision maker is designed to deal with disturbances in advance and compensate the limitation of feedback scheme of the conventional modifier adaptation. It is constructed by historical data and deep machine learning, and combined with the modifier adaptation. When disturbances occur, the decision maker provides

an initial point close to the true optimum by exploiting the historical data. As the information is accumulated, a better initial point for modifier adaptation is obtained. Constrained optimization of numerical example and run-to-run bioprocess are illustrated to validate the utility of the proposed method.

Second, there may exist a problem that estimation of the experimental gradient may not work properly for the conventional modifier adaptation scheme in the case of highly multivariate systems. In this thesis, we compare the optimization performance of gradient estimation for conventional modifier adaptation approaches and regression methods, such as multivariate linear regression, partial least squares regression, and principal component analysis. The moving average input update strategy and latent variable space model based algorithm are proposed to suppress excessive updates and improve the convergence rate and stability near the Karush-Kuhn-Tucker (KKT) point. Several simulation results of fed-batch operation of a bioreactor show that regression-based methods, especially latent variable space modeling, outperform conventional methods in the optimization of the highly multivariate system. In addition, the simulations show that both fast convergence and stability near the KKT point can be achieved by using the proposed latent variable space model-based algorithm.

Third, a novel robust structure of modifier adaptation is proposed to handle the absence of prior model information and noisy measurement. Because the newly proposed modifier adaptation has properties of convex and high order approximating formation, there exist advantages such as satisfaction of model adequacy and robustness while estimating the experimental gradient. The convergence and robustness

are proven analytically and by various simulations. In addition, during the iteration steps of the existing schemes of modifier adaptation, the input trajectory may become infeasible given plant constraints. The new formulation of the modifier adaptation ensures the input feasibility of the iterative optimization. By adjusting the coefficients of approximate model in advance, not only the input feasibility is ensured, but also the necessary conditions of optimality for plant equations are satisfied during the iterative updates. An illustrative example of constrained optimization of isothermal continuous stirred-tank reactor validates utility of the proposed method.

Keywords: Modifier adaptation, Machine learning, Model-plant mismatch, Data-driven optimization

Student Number: 2012-20974

Contents

Abstract	i
1. Introduction	1
1.1 Background and motivation	1
1.2 Literature review	3
1.2.1 Real time optimization	3
1.2.2 Optimality loss by model-plant mismatch	7
1.2.3 Methods to overcome the model-plant mismatch	8
1.3 Major contributions of this thesis	17
1.4 Outline of this thesis	19
2. Data-driven optimization via modifier adaptation	20
2.1 Introduction	20
2.2 Satisfaction of necessary conditions of optimality	22
3. Three issues of modifier adaptation	25
3.1 Issue 1: Frequent and large disturbance	25
3.1.1 Design of feedforward decision maker using machine learning and historical disturbance data	25
3.1.2 Illustrative example	45
3.1.3 Run-to-run optimization of bioprocess	57
3.1.4 Concluding remarks	63
3.2 Issue 2: Experimental gradient estimation under noisy and multivariate condition	64
3.2.1 Importance of the gradient estimation for the modifier adaptation	64

3.2.2	Motivational example: Run-to-run optimization of bioreactor	66
3.2.3	Conventional experimental gradient estimation	71
3.2.4	Regression based gradient estimation and its application to modifier adaptation	74
3.2.5	Concluding remarks	104
3.3	Issue 3: A novel structure of modifier adaptation for robustness	105
3.3.1	Feasibility and structural robustness	105
3.3.2	Proposition of new structural modifier adaptation	110
3.3.3	Illustrative example	124
3.3.4	Concluding remarks	130
4.	Conclusions and future works	131
4.1	Conclusions	131
4.2	Future works	132
	Bibliography	135

List of Tables

Table 3.1.	Nominal parameter values for numerical example in issue 1	36
Table 3.2.	Nominal parameter values for bio-process example in issue 1	62
Table 3.3.	Nominal values for plant and model equations in issue 2	70
Table 3.4.	Nominal values of the parameters in issue 3	129

List of Figures

Figure 1.1. The process of optimization and control of bioreactor example by RTO.	4
Figure 1.2. Model improvement of microalgal photobioreactor system by addition of new coefficients [51]. (a) The modelling performance for lipid concentration before the improvement, (b) The improved modelling performance for lipid concentration after adding a new coefficient in the dynamic equation.	10
Figure 1.3. Comparison of the conventional FBA model based optimization and the explicit dynamic flux balance analysis model based optimization [58].	15
Figure 1.4. Comparison of the performance of the optimized result by the explicit dynamic flux balance analysis modelling and the heuristics. A bioethanol production system is simulated for the comparison.	16

Figure 3.1. Optimal point change by the disturbance and resulting iterative optimization performances in the scheme of modifier adaptation: (a) 4 different classes of optimal point; (b) Iterative optimization result according to the starting points; (c) Objective values by iterative optimization according to the starting points; (d) Cumulative error between real optimum and sub-optimum by iterative modifier adapting optimization according to the starting points.	35
Figure 3.2. Proposed algorithm which combines the conventional modifier adaptation scheme and feed-forward decision maker by historical disturbance data and machine learning	42
Figure 3.3. Compare of the algorithms: (a) Conventional modifier adaptation without feed-forward decision maker; (b) The proposed modifier adaptation with feed-forward decision maker. . . .	44

Figure 3.4. Iterative optimization performance in immature historical data situation: (a) Historical disturbance data set in which only from 1st to 6th disturbance cases have occurred; (b) Iterative optimization results by both of the proposed method and conventional modifier adaptation; (c) Objective function errors between real optimum and sub-optimum by the proposed method; (d) Objective function errors between real optimum and sub-optimum by the conventional modifier adaptation. 47

Figure 3.5. Iterative optimization performance when the same disturbance is repeated: (a) Historical disturbance data set in which only from 1st to 6th disturbance cases have occurred (immature historical data situation); (b) Iterative optimization results by both of the proposed method and conventional modifier adaptation when the same disturbance is repeated in $k = 46$ and $k = 55$; (c) Objective function errors between real optimum and sub-optimum by the proposed method; (d) Objective function errors between real optimum and sub-optimum by the conventional modifier adaptation. 49

Figure 3.6. Iterative optimization performance in mature historical data situation: (a) Historical disturbance data set in which from 1st to 9th disturbance cases have sufficiently occurred; (b) Iterative optimization results by both of the proposed method and conventional modifier adaptation; (c) Objective function errors between real optimum and sub-optimum by the proposed method; (d) Objective function errors between real optimum and sub-optimum by the conventional modifier adaptation. 54

Figure 3.7. Iterative optimization performance by narrow structure feed-forward decision maker in mature historical data situation: (a) Historical disturbance data set in which from 1st to 9th disturbance cases have sufficiently occurred; (b) Iterative optimization results by both of the proposed method and conventional modifier adaptation; (c) Objective function errors between real optimum and sub-optimum by the proposed method having narrow structure feed-forward decision maker; (d) Objective function errors between real optimum and sub-optimum by the conventional modifier adaptation. 55

Figure 3.8. Iterative optimization performance in mature and complex historical data situation: (a) Historical disturbance data set in which from 1st to 9th disturbance cases have sufficiently occurred, but not had well-classified distribution; (b) Iterative optimization results by both of the proposed method and conventional modifier adaptation; (c) Objective function errors between real optimum and sub-optimum by the proposed method; (d) Objective function errors between real optimum and sub-optimum by the conventional modifier adaptation. 56

Figure 3.9. Iterative optimization performance in bio-process example: (a) Historical disturbance data set for constructing feed-forward decision maker; (b) Iterative optimization results by both of the proposed method and conventional modifier adaptation; (c) Objective function errors between real optimum and sub-optimum by the proposed method; (d) Objective function errors between real optimum and sub-optimum by the conventional modifier adaptation. 61

Figure 3.10. The fed-batch bioreactor system to be simulated for the comparison of gradient estimating methods. The key variables are the concentrations of biomass, substrate and product, and the working volume of the reactor. The manipulated variables are 50 discretized feed flow rates in step form. 69

Figure 3.11. Simulation results by Brdyś and Tatjewski's gradient estimation method and modifier adaptation. It is assumed that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are [50.1, 2.3] and [52.1, 1.9], respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation. 83

Figure 3.12. Simulation results by MLR based gradient estimation and modifier adaptation. It is set that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation and the number of data points for gradient estimation is 50. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are [51.8, 2.5] and [53.1, 1.5], respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation. 84

Figure 3.13. Simulation results by MLR based gradient estimation and modifier adaptation with the moving average input update strategy. It is set that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation, the number of data points for gradient estimation is 50 and the exponential filter gain for moving average input update is 0.8. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are [52.8, 1.2] and [54.7, 0.9], respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation. 85

Figure 3.14. Simulation results by MLR with the moving average input update strategy depending on the number of data points for gradient estimation. (a) Optimization performance according to the number of data points, Nw . (b) The condition numbers of each data point case along the iterative optimization. 86

Figure 3.15. Simulation results by PLSR based gradient estimation and modifier adaptation. It is set that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation and the numbers of data points for gradient estimation and principal components are 50 and 4, respectively. The moving average input update strategy is not utilized in this case. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are [55.5, 0.05] and [55.5, 0.04], respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation. 94

Figure 3.16. Simulation results by PLSR based gradient estimation and modifier adaptation with the moving average input update strategy. It is set that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation, the exponential filter gain for moving average input update is 0.8 and the numbers of data points for gradient estimation and principal components are 50 and 4, respectively. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are [54.8, 0.4] and [55.5, 0.6], respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation. 95

Figure 3.17. Simulation results by PLSR depending on the number of data points for gradient estimation. For emphasizing the comparison of the optimization performance near the KKT points, it is set that there exists normally distributed noise on each measurement having zero mean and 0.2 standard deviation. The numbers of data points for gradient estimation and principal components are 50 and 4, respectively. (a) Optimization performance when the number of data points for gradient estimation is 10. (b) Optimization performance when the number of data points for gradient estimation is 30. (c) Optimization performance when the number of data points for gradient estimation is 50. 96

Figure 3.18. Simulation results by PCR based gradient estimation and modifier adaptation. It is set that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation and the numbers of data points for gradient estimation and principal components are 50 and 10, respectively. The moving average input update strategy is not utilized in this case. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are [55.5, 0.08] and [55.5, 0.04], respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation. 97

Figure 3.19. Simulation results by PCA and PMP based gradient estimation and modifier adaptation. It is set that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation and the numbers of data points for gradient estimation and principal components are 50 and 10, respectively. The moving average input update strategy is not utilized in this case. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are [55.5, 0.04] and [55.5, 0.04], respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation. 98

Figure 3.20. Simulation results by the proposed algorithm in Section 6.3. It is set that there exists normally distributed noise on each measurement having zero mean and 0.2 standard deviation. By the proposed method, the numbers of data points for gradient estimation and principal components are varying along the iteration of optimization. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are [55.4, 0.08] and [55.5, 0.07], respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 11, 100 and 200 input updates by the modifier adaptation. (d) Optimal number of principal components when the output is objective function. (e) Optimal number of principal components when the output is the first inequality constraint. (f) Optimal number of principal components when the output is the second inequality constraint. 103

Figure 3.21. Update of objective function value by the scheme of the conventional modifier adaptation in Eq. (2.1). 108

Figure 3.22. Update of constraint functions by the scheme of the conventional modifier adaptation in Equ. (2.1). 109

Figure 3.23. Update of objective function value by the scheme of the proposed modifier adaptation in Eq. (3.68) and Eq. (3.69a-d).	122
Figure 3.24. Update of constraint functions by the scheme of the proposed modifier adaptation in Eq. (3.68) and (3.69a-d).	123
Figure 3.25. Update of objective function of the continuous stirred-tank reactor example by the scheme of the proposed modifier adaptation in Eq. (3.68) and Eq. (3.69a-d).	127
Figure 3.26. Update of constraint functions of the continuous stirred-tank reactor example by the scheme of the proposed modifier adaptation in Eq. (3.68) and Eq. (3.69a-d).	128

Chapter 1

Introduction

1.1 Background and motivation

Chemical and biological processes are characterized by complex dynamics with constraints and many possible operating modes to handle [5]. In operating these processes, enhancing productivity while satisfying the constraints is one of the key issues in global competitions [1, 11]. Thus various process optimization techniques have been introduced and applied for this purpose. Typical process optimization uses mathematical models and numerical procedures to compute an optimal solution [2]. However, the time and resources required to develop an accurate process model incurs a significant cost in general [9]. Moreover, uncertainties like exogenous disturbances and their unknown effects on the process make modelling more challenging and often lead to model-plant mismatch [7]. If this mismatch occurs, open-loop optimization cannot guarantee optimal performance and the constraints must be revised by additional control layer in most cases [8, 6]. Thus, the techniques that use measurements to offset the uncertainty and lead the process to real plant's optimum, such as real time optimization (RTO), have recently received much attention [3].

A classical two-step approach updates model parameters and performs optimization [3]. The idea is to repeatedly estimate the model parameters and use the updated model to compute a new operating point via optimization. In this method, the updated model is expected to yield a better description of the real plant near the current operating mode [5]. However, the two-step approach works well only if (1) there exists little structural model-plant mismatch and (2) the changes in operating conditions provide sufficient excitations for parameter estimation. It is known that these are difficult to satisfy in the real process [5].

Instead of updating model parameters, discrepancy between model and plant can be directly reflected in the model. A method called integrated system optimization and parameter estimation (ISOPE) incorporates a gradient modification term into the objective function to achieve convergence to a Karush-Kuhn-Tucker or KKT point that satisfies necessary condition of plant's optimality [6]. Chachuat et al. [2] recently introduced a different way to compensate model-plant mismatch by adding zero and first-order modification terms to constraints as well as to objective. This class of approaches are called modifier adaptation, where the necessary conditions of optimality of the plant and model can match [9]. The main advantage of modifier adaptation lies in its proven ability to converge to KKT point of a plant under model-plant mismatch arising from structural mismatch as well as parametric uncertainty [3]. There have been successful applications of modifier adaptation schemes to several case studies involving chemical and biological processes under uncertainty, where true plant optima could be achieved against model-plant mismatches [9, 4].

1.2 Literature review

1.2.1 Real time optimization

Typical process optimization and its implementation have a sequential structure. First, optimization is performed for whole plant (scheduling) and for each process unit. Real time optimization (RTO) usually means the optimization for each process unit, and it is known as a key step for improving the efficiency or productivity of chemical or biological process in real industry [68]. It has been applied to several advanced processes such as continuous stirred-tank reactor (CSTR) or fed-batch reactor system [66, 64]. The solution of RTO can be a steady state set-point for CSTR or time-varying trajectory for the fed-batch system. After the optimization step, control layer is activated to make the process unit operate as the solution of RTO step. Model predictive controller (MPC) is operated as a master controller for whole system having constraints and multiple variables, and proportional-integral-derivative (PID) controllers are operated as sub-controllers for each unit [65, 67]. The whole process of the RTO by an example of bioreactor system is illustrated in Figure 1.1.

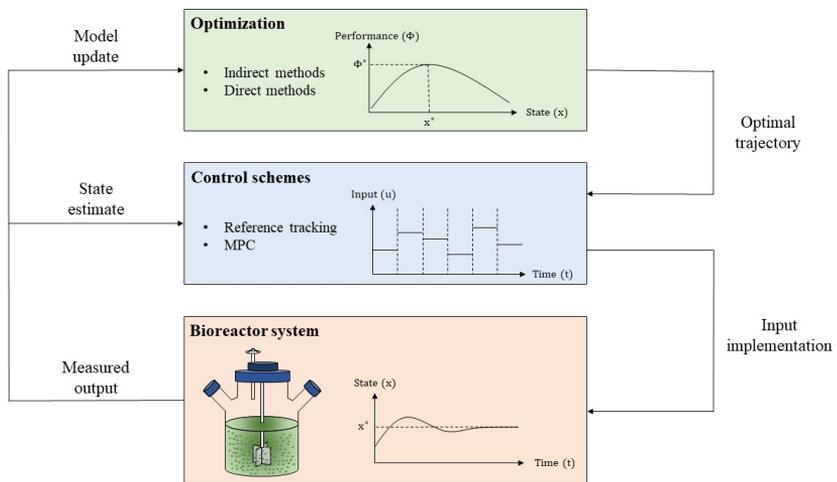


Figure 1.1: The process of optimization and control of bioreactor example by RTO.

Typical RTO in real industry is based on the numerical model. The model includes the first principle based ones, the numerical ones based on process data, and the hybrid ones. An example of first principle based modelling is mass or energy balanced equations in the chemical reactor design. Examples of the data based modelling are neural network and regression methods such as partial least squares. The hybrid modelling is a combinational model of the first principle based and data based ones. An example of the hybrid modelling is a fed-batch reactor case that the dynamic model structure is based on the mass balance, and the reaction rates are based on the neural network. It is also referred to as grey box model, while the first principle based modelling is white box and the data based modelling is black box.

On the other hand, the models for RTO can be divided into steady-state system or dynamic system. Examples of steady-state system in chemical or biological processes are CSTR or chemo-stat reactor. Because the equations of reaction rate in this reactors usually contain nonlinearity, the optimization problem becomes a nonlinear programming (NLP). Then it is solved by using NLP algorithm such as generic algorithm (GA) for global optimization, or sequential quadratic programming (SQP) for searching a local optimal point [62, 63]. In typical RTO, the local optimization methods are chosen because the computational load may be too large to use the global optimization approaches. Thus, problems related to the local optimization methods such as selection of the initial point can exist in application of the RTO implementation to the real process.

An example of dynamic system optimization is fed-batch bioreactor. Operational optimization of bioreactor typically refers to cal-

calculation of optimal trajectories of time-varying inputs, typically feed rate or temperature, for achieving prescribed objective such as maximization of specific product's yield. Being different from the steady-state optimization case, it consists of differential algebraic equations (DAEs) that describe the dynamic behavior of the system, typically mass balance and energy balance equations given as ordinary differential equations and algebraic equations ensuring their physical and thermodynamic relations [59, 60]. Although there are several ways to solve the dynamic optimization according to the characteristics of the system, they can be summarized into two categories; indirect methods and direct methods [59, 60]. Direct methods transform the original dynamic optimization problem into a large-sized NLP by discretizing control and state variables, and then solve it using specialized NLP algorithm such as SQP [61]. According to the way of discretization, they can be further classified into a sequential method or a simultaneous method. Unlike the indirect approaches which aims to find an analytical solution, a numerical solution is obtained via direct approaches. Even though this solution is an approximate one, the benefit of utilizing direct approaches comes from the fact that one can always obtain the solution even when the number of variables and constraints is very large.

1.2.2 Optimality loss by model-plant mismatch

Consider the plant equation based optimization, which is based on the steady-state model or discretized dynamic system's model:

$$\begin{aligned} u_p^* &:= \arg \min_u \Phi_p(u) \\ \text{s.t. } & G_p(u) \leq 0 \end{aligned} \quad (1.1)$$

where $\Phi_p \in \mathbb{R}$, $G_p \in \mathbb{R}^{1 \times m}$ and $u \in \mathbb{R}^{n \times 1}$ are the plant's objective, constraint equations and input variables. n and m are the numbers of input variables and constraint equations and \mathbb{R} denotes the real number set. The KKT conditions or the necessary conditions for optimality are:

$$G_p(u_p^*) \leq 0 \quad (1.2a)$$

$$\nabla \Phi_p(u_p^*) + (v_p^*)^T \nabla G_p(u_p^*) = 0 \quad (1.2b)$$

$$v_p^* \geq 0 \quad (1.2c)$$

$$v_p^{*T} G_p(u_p^*) = 0 \quad (1.2d)$$

where $v_p^* \in \mathbb{R}^{m \times 1}$ is the Lagrange multiplier for the plant equations. On the other hand, model equation based optimization is formulated as:

$$\begin{aligned} u_m^* &:= \arg \min_u \Phi_m(u) \\ \text{s.t. } & G_m(u) \leq 0 \end{aligned} \quad (1.3)$$

where $\Phi_m (\neq \Phi_p)$ and $G_m (\neq G_p)$ are the model's objective and constraint equations, respectively. The KKT conditions of model equa-

tions are given differently from the plant's ones:

$$G_m(u_m^*) \leq 0 \quad (1.4a)$$

$$\nabla \Phi_m(u_m^*) + (v_m^*)^T \nabla G_m(u_m^*) = 0 \quad (1.4b)$$

$$v_m^* \geq 0 \quad (1.4c)$$

$$v_m^{*T} G_m(u_m^*) = 0 \quad (1.4d)$$

where $v_m^* \in \mathbb{R}^{m \times 1}$ is the Lagrange multiplier for model equations. In the presence of model-plant mismatch, the values and their gradients predicted by the model equations are not equal to those of the plant equations. Therefore, the optimal solution by the model equations cannot satisfy the necessary conditions of optimality for plant equations Eq. (1.2a-d). Thus, open-loop implementation of the solution to Eq. (1.3) might lead to a suboptimal operating point, and even worse, infeasible solution may be implemented from the incorrect model.

1.2.3 Methods to overcome the model-plant mismatch

In order to overcome this optimality loss problem by the model-plant mismatch, several methods have been proposed. First, there have been studies focusing on the improvement of the model itself. The modifications can be implemented in several ways such as addition of new coefficients or utilization of intracellular information. In a recent study by Yoo et al. [51], the authors observed the mismatch between the model prediction and the operation data, and concluded that it was owing to the assumption that the yield coefficients Y_{XQ} and Y_{XL} are held constant throughout the process. Hence, they mod-

ified the model by introducing time-varying yield coefficient. As a result of the modification, the model showed an improved prediction capability in Figure 1.2. The authors also use the modified model to optimize the bioreactor system using nonlinear model predictive control scheme [52].

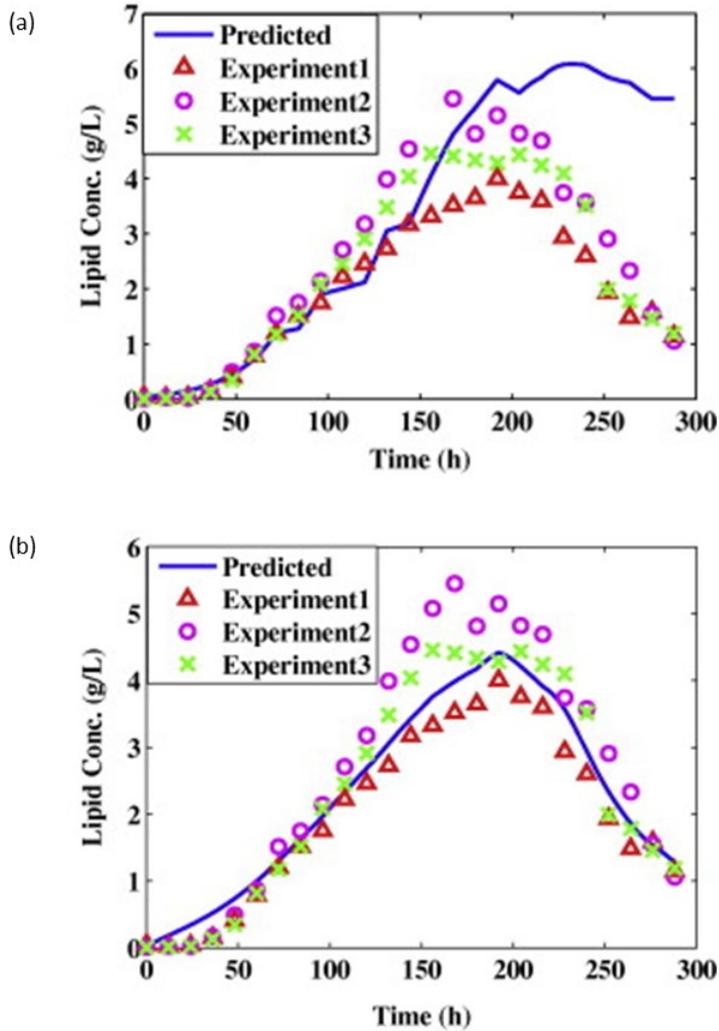


Figure 1.2: Model improvement of microalgal photobioreactor system by addition of new coefficients [51]. (a) The modelling performance for lipid concentration before the improvement, (b) The improved modelling performance for lipid concentration after adding a new coefficient in the dynamic equation.

Another example of model improvement is a case that intracellular metabolic information is utilized to construct the dynamic model. Flux balance analysis (FBA) is a technique that estimates the intracellular reaction rates of a specific organism [53]. It is based on a stoichiometric model of intracellular metabolism with steady state assumption and yields intracellular dynamics as a set of linear algebraic equations. In solving this intracellular dynamics, FBA is formulated as a linear programming (LP) problem with biological functions such as maximizing the production of biomass as its objective function [53, 54, 55]:

$$\frac{dX}{dt} = Sv \quad (S \in \mathbf{R}^{n \times m}, v \in \mathbf{R}^n, X \in \mathbf{R}^m) \quad (1.5a)$$

$$lb_i \leq v_i \leq ub_i \quad (i = 1, \dots, n) \quad (1.5b)$$

where X is the vector of masses for m metabolites, S is the stoichiometric matrix describing mass balances, and v_i is the flux for reaction i (change of mass or mole per time per dry-weight of biomass) with its lower and upper bounds, lb_i and ub_i , respectively. Since it is assumed that the transients inside the metabolism are much faster than the dynamics of extracellular environment, the intracellular metabolic network are often assumed to be at a quasi-steady state [70]. Thus the mass balance equations become algebraic equations as $Sv = 0$ ($S \in \mathbf{R}^{m \times n}, v \in \mathbf{R}^n$). Since the number of intracellular reactions is generally larger than that of metabolites, i.e., $n > m$, it is an undetermined problem and the fluxes are the null space of the existing equality equations with the bounds. In order to specify the flux values, FBA introduces biochemical objectives of the organism's metabolism such as maximizing the biomass producing flux, maximizing the ATP uti-

lization, or minimizing the number of reaction steps [73]. Then the flux values can be obtained by solving the following optimization problem:

$$\max f_o(v) \quad (1.6a)$$

$$\text{subject to } Sv = 0 \quad (1.6b)$$

$$lb_i \leq v_i \leq ub_i (i = 1, \dots, m) \quad (1.6c)$$

$$S \in \mathbf{R}^{n \times m}, v \in \mathbf{R}^m, X \in \mathbf{R}^n \quad (1.6d)$$

where $f_o(v)$ is the cellular objective function, which is often expressed as a linear combination of the fluxes, i.e. $c^T v$. The mathematical characteristics of the optimization problem (and its corresponding solution) is determined by the objective function $f_o(v)$. Since maximizing the biomass production rate has been selected as the cellular objective function in several studies [74, 53], this work also uses the same objective function. With this metabolism-incorporated modelling framework, organism-specific models with larger valid description ranges than the empirical one can be constructed. Dynamic flux balance analysis (DFBA) is a way of modelling that connects the reactor's macroscopic dynamics to the organism's microscopic metabolism through FBA. It can predict dynamics of substrates, biomass and product concentrations in batch or fed-batch cultures. This method assumes that metabolite concentrations rapidly equilibrate in response to extracellular perturbations [69]. This is acceptable because the intracellular metabolic reactions are much faster in order of magnitudes than the extracellular reaction rates such as growth kinetics [54]. DFBA models are obtained by combining the optimization

(FBA) for intracellular reaction rates with dynamic mass balances on key extracellular substrates and products, often in the form of ordinary differential equations [70]. The intracellular and extracellular phenomena are bridged through the cellular growth rate and substrate uptake kinetics [69]. A typical formulation is given as

$$\frac{dX}{dt} = f_X(\mu, X) \quad (1.7a)$$

$$\frac{dC_j}{dt} = f_{C_j}(U_j, S_j, X) \quad (j \in J) \quad (1.7b)$$

$$U_k = f_k(C_1, \dots, C_x) \quad (k \in K \subset J) \quad (1.7c)$$

$$S_l = f_l(C_1, \dots, C_y) \quad (l \in L \subset J) \quad (1.7d)$$

$$\{\mu, U_{k'}, S_{l'} | k' \notin K, l' \notin L\} \in \arg \max_v c^T v \quad (1.7e)$$

$$\text{subject to } Sv = 0 \quad (1.7f)$$

$$lb_i \leq v_i \leq ub_i \quad (i = 1, \dots, m) \quad (1.7g)$$

$$S \in \mathbf{R}^{n \times m}, v \in \mathbf{R}^m, X \in \mathbf{R}^n \quad (1.7h)$$

where U_i and S_i are the uptake and secretion rates of a key substrate i , respectively, μ is the growth rate, X is the biomass concentration, and C_j is the concentration of substrate j . In order to simulate the dynamics of organism's metabolism with suggested model formulation, it first discretizes the time horizon and solves the linear program, that is, FBA with the uptake rates at each time point. Then, as the optimized solutions, the growth and secretion rates are obtained from the FBA. The extracellular substrate's dynamics are integrated with the obtained linear program solutions over a short time period to obtain the concentrations at the next time point. The same procedure is repeated until the final time point. This approach is referred to as

the static optimization approach (SOA) [71]. Varma and Palsson suggest a wild type *E. coli*'s flux balance model to predict the cellular growth and by-product secretion [74]. Specifically, biomass, acetate, formate and ethanol secretions are predicted under anaerobic batch culture with glucose as a carbon source. Mahadevan et al. study the diauxic growth of *E. coli* based on FBA [71]. This study simplifies the metabolic network of *E. coli* using extreme pathways and suggests a dynamic optimization approach for simulating the dynamics of biomass, glucose, oxygen, and acetate only [72].

Several researches on modelling and operational applications using DFBA models have been reported [56, 57]. However, the DFBA model has a structural problem that it has an embedded optimization in every step of integration, linear programming to estimate intracellular reaction rates given a different set of boundary conditions. This embedded optimization steps poses a computational challenge in implementing the model-based approaches to calculating an optimal feeding strategy. To address the issue of DFBA modelling, it is suggested to solve the embedded linear program via parametric programming in a recent study by Jeong et al. [58]. The obtained explicit solution sets are the estimates for intracellular reaction rates by FBA and combined with kinetic model. This new DFBA modelling framework referred to as explicit DFBA or xDFBA does not involve any embedded linear program. Thus, this can be utilized in various model based applications without much computational burden. However, these effort require a lot of resources, and as a result, it is virtually impossible to construct a complete model matching perfectly with the plant equation.

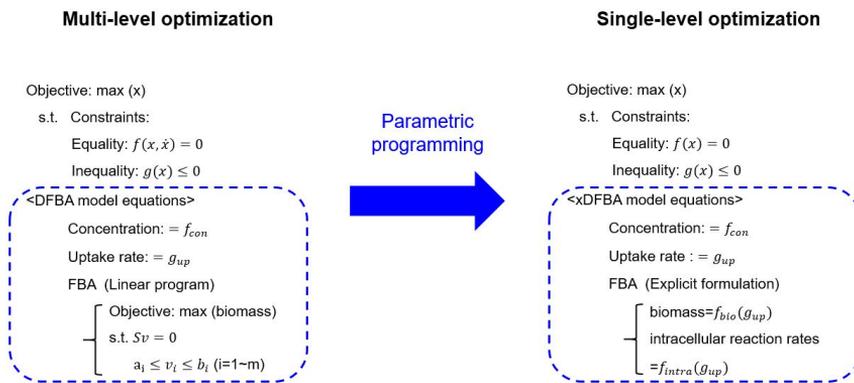


Figure 1.3: Comparison of the conventional FBA model based optimization and the explicit dynamic flux balance analysis model based optimization [58].

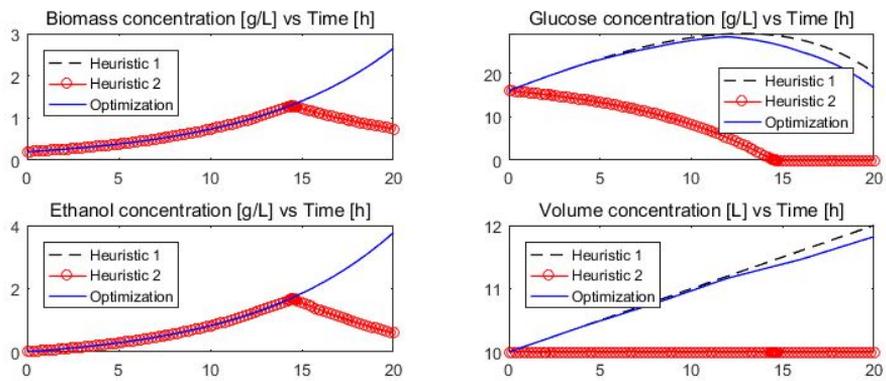


Figure 1.4: Comparison of the performance of the optimized result by the explicit dynamic flux balance analysis modelling and the heuristics. A bioethanol production system is simulated for the comparison.

Second, there exist an approach updating the model parameters iteratively based on the measurement. This method is referred to as two-step approach as it consists of separated steps: parameter estimation and optimization. It has been widely used in general industrial process RTOs. However, there is no guarantee that an exact model matching the plant will be found through parameter updates and the operating condition can be sufficiently changed to provide enough variation for parameter estimation. Moreover, if the model shape is poorly defined in the first place and structural mismatch exists, the problem becomes more serious.

Third, data-driven optimization methods are proposed to overcome these limitations. Initially, integrated system optimization and parameter estimation (ISOPE) was proposed to converge the solution to a KKT point by modifying the gradient of the model objective [6]. Recently, a new scheme called modifier adaptation was proposed and applied to several optimization problems. This method updates the model structure by adding the error between the measurement and model prediction, called zeroth- and first-order modification terms both to the objective and constraints [1]. It has been proven to converge to the plant KKT point for all kinds of model-plant mismatches, including large structural ones, for which two-step approaches generally fail [3]. This thesis focuses on research to solve various issues arising from data-based optimization using the modifier adaptation.

1.3 Major contributions of this thesis

In this thesis, 3 issues of data-driven optimization named modifier adaptation are discussed. First, the decrease of optimization per-

formance by repeated disturbances is studied. As the conventional modifier adaptation scheme has only a feed-back function to optimize the process, it shows slow adapting performance to the frequent and repeated disturbance. In order to solve this problem, this thesis proposes a feed-forward decision maker design by the historical disturbance data and machine learning called deep neural network. With this feed-forward decision maker which gives a good starting point of manipulated variable according to the disturbance information, faster management is possible and the optimization performance is improved. Second, efficient estimating methods for experimental gradient are discussed, especially when the number of manipulated variable is large. Various regression techniques are applied to the estimation of experimental gradient and their efficiency are compared. Among the regression methods, latent variable space modelling approaches such as partial least squares show the best estimating performance. In addition, moving average gradient estimation rule is newly proposed and its satisfaction of plant KKT conditions upon convergence is proven. Third, a novel robust structure of modifier adaptation is proposed to handle the absence of prior model information and noisy measurement. Because the newly proposed modifier adaptation has properties of convex and high order approximating formation, there exist advantages such as satisfaction of model adequacy and robustness while estimating the experimental gradient. The convergence and robustness are proven analytically and by various simulations. Each of these issues and solutions will be explained in detail in the following Chapters.

1.4 Outline of this thesis

The remainder of the thesis is organized as follows. Chapter 2 introduces the implementation of modifier adaptation to the process under model-plant mismatch and the proof about satisfaction of necessary conditions of optimality or KKT conditions of plant by the modifier adaptation. In Chapter 3, several issues are dealt with while applying the modifier adaptation to the industrial process. First, an issue of decreasing the optimization performance by the repeated disturbance is discussed and solved. Second, the efficient way of experimental gradient estimation when existing large number of manipulated variables is studied. Third, proposition of new structure of modifier adaptation to handle the absence of prior model information and noisy measurement. Lastly, Chapter 4 provides concluding remarks and future works.

Chapter 2

Data-driven optimization via modifier adaptation

2.1 Introduction

Modifier adaptation (or iterative gradient-modification optimization) uses the plant measurements to update the model Eq. (1.3) and computes the modified optimization problem iteratively [6, 3]. At each iteration, the zero and first-order modifiers, which are the differences between the predicted and measured values of the constraints and gradients, are updated. These modifiers add input-affine corrections to the model Eq. (1.3). The zeroth-order modifiers allow for satisfying the feasibility conditions of the plant, and the first-order modifiers are essential for satisfying the KKT conditions. Thus, gradients of the plant equations should be estimated with a sufficiently high level of accuracy for the success of modifier adaptation [3] in such a way that the necessary conditions of optimality of the modified optimization problem match those of the plant Eq. (1.2) upon convergence [5]. The conventional modifier adaptation scheme is given by [5]:

$$\begin{aligned}
u^*(k+1) &:= \arg \min_u \Phi_m(u) := \Phi(u) + \delta^{\Phi,imp}(k) + \lambda^{\Phi,imp}(k)(u - u^*(k)) \\
\text{s.t. } G_m(u) &:= G(u) + \delta^{G,imp}(k) + \lambda^{G,imp}(k)(u - u^*(k)) \leq 0.
\end{aligned} \tag{2.1}$$

where $u^*(k+1) \in \mathbb{R}^{n_u}$, $u^*(k) \in \mathbb{R}^{n_u}$, $\Phi : \mathbb{R}^{n_u} \rightarrow \mathbb{R}$, $G_i : \mathbb{R}^{n_u} \rightarrow \mathbb{R}$ (with $i = 1, \dots, n_G$), $\Phi_m : \mathbb{R}^{n_u} \rightarrow \mathbb{R}$, $G_{m,i} : \mathbb{R}^{n_u} \rightarrow \mathbb{R}$, $\delta^{\Phi,imp} \in \mathbb{R}$, $\delta_i^{G,imp} \in \mathbb{R}$ (with $i = 1, \dots, n_G$), $\lambda^{\Phi,imp} \in \mathbb{R}^{n_u}$ and $\lambda_i^{G,imp} \in \mathbb{R}^{n_u}$ (with $i = 1, \dots, n_G$) are the solution of modifier adaptation implemented at $k+1$ and k , model equations of the objective and constraints, modified equations of the objective and constraints and the implemented zeroth- and first-order modifiers, respectively. The zeroth- and first-order modifiers for implementation are defined [5]:

$$\begin{aligned}
\delta^\Phi(k) &:= \Phi_p(u^*(k)) - \Phi(u^*(k)) \\
\delta^G(k) &:= G_p(u^*(k)) - G(u^*(k)) \\
\lambda^\Phi(k) &:= \nabla \Phi_p(u^*(k)) - \nabla \Phi(u^*(k)) \\
\lambda^G(k) &:= \nabla G_p(u^*(k)) - \nabla G(u^*(k)) \\
\Lambda(k) &:= [\delta^\Phi(k), \delta^G(k), \lambda^\Phi(k), \lambda^G(k)]^T \\
\Lambda^{imp}(k) &= (1 - \beta)\Lambda^{imp}(k-1) + \beta\Lambda(k) \\
&= [\delta^{\Phi,imp}(k), \delta^{G,imp}(k), \lambda^{\Phi,imp}(k), \lambda^{G,imp}(k)]^T
\end{aligned} \tag{2.2}$$

where $\delta^\Phi \in \mathbb{R}$, $\delta_i^G \in \mathbb{R}$ (with $i = 1, \dots, n_G$), $\lambda^\Phi \in \mathbb{R}^{n_u}$, $\lambda_i^G \in \mathbb{R}^{n_u}$ (with $i = 1, \dots, n_G$), $\Lambda \in \mathbb{R}^{2n_G+2}$, $\Lambda^{imp} \in \mathbb{R}^{2n_G+2}$ and $\beta \in \mathbb{R}^{2n_G+2}$ are the zeroth- and first-order modifiers by subtracting the model and real values of objective and constraints and its gradients, matrix of the calculated and implemented modifiers, and exponential filter gain, re-

spectively. The exponential filter suppresses the excessive input update and reduces the influence of noise. It is known that the zeroth-order modifier for the objective function ($\delta^{\Phi, imp}(k)$) does not affect the optimality, thus in recent studies, this term is eliminated in the scheme of modifier adaptation [42, 75].

2.2 Satisfaction of necessary conditions of optimality

For satisfying the KKT conditions of plant equation by the modifier adaptation, a property called model adequacy should be satisfied. The model adequacy for the modifier adaptation is defined as follows [5]:

Definition 2.1. *Let u_p^* be the plant's local optimal point and it satisfies the constraints. If the process model satisfies a condition that the reduced Hessian of the cost function Φ is positive definite at u_p^* ,*

$$\nabla_r^2 \Phi(u_p^*) > 0 \quad (2.3)$$

then the process model is adequate for the modifier adaptation.

It is known that the model adequacy is a necessary condition for convergence to a KKT point in the modifier adaptation scheme [5]. For this, a modifier adaptation making a model structure convex was proposed by Marchetti et al. [5]. In this thesis, a method is proposed a modifier adaptation with structural robustness and feasibility, moving one step further from convexity (Chapter 3.1).

The most attractive property of the modifier adaptation is that the resulting KKT point of Eq. (2.1) satisfies the necessary conditions of optimality of plant in Eq. (1.2a-d), upon convergence. In other words,

it forces the KKT points of the model-based optimization to match those of the plant optimization, and this advantage cannot be achieved via a classical two-step approach [5].

Theorem 2.1. *Let the gain matrix β in Eq. (2.2) be nonsingular and assume that the modifier adaptation algorithm converges to a point $\lim_{k \rightarrow \infty} u_k = u_\infty$. This point satisfies the KKT conditions of modified equation Eq. (2.1). Then, u_∞ satisfies the KKT conditions of plant in Eq. (1.2a-d).*

Proof This proof is quoted from the Marchetti et al. [5]. As β is nonsingular, the following equations for zeroth and first-order modifiers are satisfied when the input is converged to a point u_∞ :

$$\begin{aligned}\varepsilon_\infty^{G_i} &= G_{p,i}(u_\infty) - G_i(u_\infty), \quad i = 1, \dots, n_g \\ \lambda_\infty^{G_i} &= \frac{\partial G_{p,i}}{\partial u}(u_\infty) - \frac{\partial G_i}{\partial u}(u_\infty), \quad i = 1, \dots, n_g \\ \lambda_\infty^\Phi &= \frac{\partial \Phi_p}{\partial u}(u_\infty) - \frac{\partial \Phi}{\partial u}(u_\infty)\end{aligned}\tag{2.4}$$

where n_g is the number of constraint equations. It is then readily seen than, upon convergence, the KKT elements for the modified problem match the corresponding elements of the modifiers for the plant,

$$\begin{aligned}G_m(u_\infty) &= G(u_\infty) + \varepsilon_\infty^{G_i} = G_p(u_\infty), \quad i = 1, \dots, n_g \\ \frac{\partial G_{m,i}}{\partial u}(u_\infty) &= \frac{\partial G}{\partial u}(u_\infty) + \lambda_\infty = \frac{\partial G_{p,i}}{\partial u}(u_\infty), \quad i = 1, \dots, n_g \\ \frac{\partial \Phi_m}{\partial u}(u_\infty, \theta) &= \frac{\partial \Phi}{\partial u}(u_\infty) + \lambda_\infty = \frac{\partial \Phi_p}{\partial u}(u_\infty)\end{aligned}\tag{2.5}$$

Because u_∞ is a KKT point for the modified problem, it satisfies

the plant KKT conditions with the associated Lagrange multipliers v_∞ . Thus, the converged point u_∞ satisfies the KKT conditions of plant with the same Lagrange multipliers. ■

It should be noted that the satisfaction of KKT conditions by the modifier adaptation does not guarantee of eliminating the optimality loss perfectly. As the satisfaction of KKT conditions is a necessary condition for the optimality, there may exist the optimality loss in spite of the modifier adaptation. In the case that the plant equation is convex, it can be sufficient condition for the optimality. This case is, however, not general in the real process environment including non-linearity. Thus, the user should know the limitation of the modifier adaptation before applying it to the process for data driven optimization. In many simulation cases, the satisfaction of the KKT conditions of plant is enough for the optimization, though they are just necessary conditions. All of the simulation cases in the following chapters show the same results. It can be interpreted that the plant equation is locally convex nearby the optimal point.

Chapter 3

Three issues of modifier adaptation

3.1 Issue 1: Frequent and large disturbance

3.1.1 Design of feedforward decision maker using machine learning and historical disturbance data

As introduced in the last Chapter, the main advantage of modifier adaptation lies in its proven ability to converge to KKT point of a plant under model-plant mismatch arising from structural mismatch as well as parametric uncertainty [3]. In real processes, however, disturbances may occur frequently and their magnitudes can be large enough to shift the process far from an optimal point. Although the conventional modifier adaptation can handle disturbances to a certain extent, it may not adequately deal with the large changes of optimal point due to the disturbance because it adjusts the model-based optimization using the plant output measurements in a post-processing fashion like feedback control. Therefore, such disturbances need to be handled proactively with a feedforward structure to provide optimal operating points in a timely fashion.

With recent advances in sensor technology, a large amount and different kinds of data are available and bring tremendous opportuni-

ties for various applications. Moreover, it is also shifting paradigms in many areas towards data-driven discovery [18]. As the amount of available data increases, machine learning that utilizes data and provides insightful analysis has become a key technique [18]. It has already been utilized in various fields such as content filtering on social networks and recommendation system on e-commerce websites [16]. It can also be used for identifying images, transcribing sound data, matching news with users' interests [22].

Various types of machine learning techniques have been introduced and improved for these purposes. The first one is support vector machine or SVM. It has advantages of low computational cost and accessible optima due to convex quadratic optimization. It is proven that SVM can effectively tackle problems with small samples, and has been utilized in many fields such as artificial intelligence, pattern recognition and machine learning [12]. However, there exists a remaining problem that the computational complexity grows exponentially with the number of training samples [12]. Artificial neural networks or ANN is an overlapped structure of neural units which mimics the way of a brain to solve problems with large clusters of neurons connected by axons. The most common types of ANN are multi-layer perceptron and radial basis function networks [21]. It has been popularly used for function approximation and pattern recognition [21]. Despite the successful applications of SVM and ANN, however, they have a disadvantage that the interpretation capability still remains a major problem for complex systems [15, 16, 12]. This limitation is caused by no allowance for latent variable subspace [12]. Especially, ANN suffers from an uncontrolled convergence speed and local optima. Moreover, parameters of ANN with

more than two layers are difficult to optimize using traditional gradient descent schemes [15]. In order to overcome these limitations, it was suggested to pre-process data by unsupervised learning that eliminates the noise and possibly reduces its searching dimensionality [13, 15]. This has opened up a new area of machine learning, called deep learning.

In contrast to the conventional machine learning techniques having shallow architectures, the deep learning refers to a class of machine learning that uses supervised and/or unsupervised strategies to automatically learn hierarchical representations in deep architectures [18]. One of the earliest deep learning schemes, the deep belief network by Hinton, can be summarized as follows: first, pre-train one layer at a time by unsupervised data and restricted Boltzmann machine to preserve information from the input; then do fine-tuning the whole network with respect to the criterion of interest [13]. This approach was shown to perform better than those trained exclusively with back-propagation [15]. In this way, machine learning can contain a deep structure of more than two hidden layers, which allows for expressing complex relationships [17]. Since the deep belief network was introduced, deep learning technology has evolved to an economically viable level in many aspects and is being utilized in many industrial applications including MNIST handwriting challenge [23], face detection [24], speech recognition [25], natural language processing [26] and soft sensor [12]. Global companies have heavily invested in research related to deep learning; Apple's Siri, Google's translator, street view and image search engine, Android's voice recognition, Facebook's recommended friends and news feed system, etc. [18].

Thanks to such advances in machine learning technology, we

propose a modifier adaptation scheme combined with feedforward decision maker to handle the issue of disturbances. This utilizes historical data of process and deep learning techniques. In this decision maker, the disturbance information in the past serves as an input and the suboptimal or optimal strategy against the disturbance serves as an output. A starting point for modifier adaptation, expected to be near the plant optimal point, is determined in a feedforward manner for each of disturbances. By starting the update near the plant optimal point with the feedforward decision maker, optimization performance is expected to improve compared with conventional modifier adaptation which handles disturbances only in a post-processing sense. In addition, continuous updates of learning model allow for prompt and optimal handling of the same disturbance experienced before.

Providing that there occurs no additional disturbance during the iterative optimization and the model-adequacy condition is satisfied [1, 76], the modifier adaptation can find a KKT point under an arbitrary model-plant mismatch as shown in Section 2. In real operational optimization problems, however, changes of disturbance may affect the process and thus change the real plant optimum significantly.

In the case of the conventional modifier adaptation scheme, the simplest strategy for handling this situation is to restart the iterative optimization at the operating point determined in the last step because it works only in a feedback or post-processing manner. Eq. (3.1a-c) show the potential issue of this strategy when a large change in

disturbance occurs at the current iteration index, k :

$$\|u(k-1) - u^*(d(k))\| > \delta_u \quad (3.1a)$$

$$u_{conv}(k) = u(k-1) \quad (3.1b)$$

$$\|\Phi(u_{conv}(k), d(k)) - \Phi(u^*(d(k)))\| > \delta_\Phi \quad (3.1c)$$

where $u_{conv}(k)$, $u(k-1)$, u^* , $d(k)$, Φ , δ_u and δ_Φ are the initial operating point by the conventional strategy, operating point in the previous step, optimal operating point, large disturbance, objective function value and arbitrary large numbers of operating point and objective value, respectively. Because of the disturbance, $u(k-1)$ becomes largely different from $u^*(d(k))$ of Eq. (3.1a). Therefore, the conventional strategy having the feedback mechanism only as in Eq. (3.1b) degrades the optimization performance as in Eq. (3.1c). Although the conventional modifier adaptation can find optimal points sequentially for each disturbance with feedback information, it is inevitable to degrade the performance due to the error in the initial point estimate over the subsequent runs of optimization. On the other hand, this performance loss can be circumvented by estimating a better initial operating point by using the feedforward decision maker:

$$\|f_d(k) - u^*(d(k))\| < \delta_u \quad (3.2a)$$

$$u_{modif}(k) = f_d(k) \quad (3.2b)$$

$$\|\Phi(u_{modif}(k), d(k)) - \Phi(u^*(d(k)))\| < \delta_\Phi \quad (3.2c)$$

where $u_{modif}(k)$ and $f_d(k)$ are the initial operating point by the modified method and feedforward decision maker dealing with disturbance

changes, respectively. If we can find a good initial operating point having relatively small error with the feedforward decision maker (3.2a) and use it as an adapted starting point (3.2b), performance enhancements not only at the starting point (3.2c) but also throughout the entire operation are expected compared to the conventional strategy in Eq. (3.1a-c). Details about designing the feedforward decision maker will be described in the following contents.

In order to motivate the importance of start point on the modifier adaptation, let's consider a numerical example:

(Disturbed plant equation)

$$\begin{aligned}
& \min_u \quad \Phi(u, d) \\
& \text{s.t.} \quad \Phi = a_1(u_1 - u_1^*)^2 + a_2(u_2 - u_2^*)^2 \\
& \quad \quad u_1^* = k_{11}d_1 + k_{21}d_2 + k_{31}d_3 + k_{41} \\
& \quad \quad u_2^* = k_{12}d_1 + k_{22}d_2 + k_{32}d_3 + k_{42} \\
& \quad \quad a_3u_1 + a_4u_2^2 \leq a_5 \\
& \quad \quad lb_1 \leq u_1 \leq ub_1 \\
& \quad \quad lb_2 \leq u_2 \leq ub_2
\end{aligned} \tag{3.3}$$

(Model equation)

$$\begin{aligned}
& \min_u \quad \Phi^m(u, d) \\
& \text{s.t.} \quad \Phi^m = a_1^m(u_1 - u_1^{m*})^2 + a_2^m(u_2 - u_2^{m*})^2 \\
& \quad \quad a_3^m u_1 + a_4^m u_2^2 \leq a_5^m \\
& \quad \quad lb_1 \leq u_1 \leq ub_1 \\
& \quad \quad lb_2 \leq u_2 \leq ub_2
\end{aligned} \tag{3.4}$$

where $u = \{u_1, u_2\}$ and $d = \{d_1, d_2, d_3, k\}$ are the manipulated variables and disturbances, respectively. The nominal parameter values for plant and model equations are presented in Table 3.1. The real optimum of plant Eq. (3.3) change in accordance with the disturbance which has following scenario:

$$\begin{aligned}
 k &= [k_{11}, k_{21}, k_{31}, k_{41}, k_{12}, k_{22}, k_{32}, k_{42}] \\
 [d_1, d_2, d_3, k] &= \mathbf{D}(\text{Case}) \\
 \text{Case} &= \text{Randi}([1, 9])
 \end{aligned} \tag{3.5}$$

$$\mathbf{D}(1) = \begin{cases} d_1 = \text{Uniform}([90, 100]) \\ d_2 = \text{Uniform}([0, 10]) \\ d_3 = \text{Uniform}([90, 100]) \\ k = [0.01, -0.15, 0.04, -3, 0.03, 0.08, -0.09, 6.3] \end{cases} \tag{3.6}$$

$$\mathbf{D}(2) = \begin{cases} d_1 = \text{Uniform}([90, 100]) \\ d_2 = \text{Uniform}([90, 100]) \\ d_3 = \text{Uniform}([90, 100]) \\ k = [0.1, 0.05, 0.05, -10, -0.07, -0.04, -0.09, 28] \end{cases} \tag{3.7}$$

$$\mathbf{D}(3) = \begin{cases} d_1 = \text{Uniform}([90, 100]) \\ d_2 = \text{Uniform}([0, 10]) \\ d_3 = \text{Uniform}([0, 10]) \\ k = [0.05, 0.07, 0.08, 3.5, -0.07, -0.06, 0.07, 7.6] \end{cases} \tag{3.8}$$

$$D(4) = \begin{cases} d_1 = \text{Uniform}([90, 100]) \\ d_2 = \text{Uniform}([90, 100]) \\ d_3 = \text{Uniform}([0, 10]) \\ k = [-0.15, 0.02, 0.03, 13.2, -0.07, 0.08, -0.05, 0.3] \end{cases} \quad (3.9)$$

$$D(5) = \begin{cases} d_1 = \text{Uniform}([0, 10]) \\ d_2 = \text{Uniform}([0, 10]) \\ d_3 = \text{Uniform}([90, 100]) \\ k = [-0.1, 0.05, -0.05, 6, 0.02, 0.09, 0.09, -0.1] \end{cases} \quad (3.10)$$

$$D(6) = \begin{cases} d_1 = \text{Uniform}([0, 10]) \\ d_2 = \text{Uniform}([90, 100]) \\ d_3 = \text{Uniform}([90, 100]) \\ k = [0.1, -0.03, -0.07, 18, 0.09, 0.04, -0.07, 3.4] \end{cases} \quad (3.11)$$

$$D(7) = \begin{cases} d_1 = \text{Uniform}([0, 10]) \\ d_2 = \text{Uniform}([0, 10]) \\ d_3 = \text{Uniform}([0, 10]) \\ k = [0.05, -0.1, 0.05, 9, -0.03, 0.1, -0.07, 9] \end{cases} \quad (3.12)$$

$$D(8) = \begin{cases} d_1 = \text{Uniform}([0, 10]) \\ d_2 = \text{Uniform}([90, 100]) \\ d_3 = \text{Uniform}([0, 10]) \\ k = [-0.06, -0.08, -0.06, 9.2, 0.1, -0.03, -0.07, 11.7] \end{cases} \quad (3.13)$$

$$D(9) = \begin{cases} d_1 = \text{Uniform}([0, 100]) \\ d_2 = \text{Uniform}([40, 60]) \\ d_3 = \text{Uniform}([40, 60]) \\ k = [0.01, 0.03, 0.02, 6, -0.01, -0.04, -0.01, 4] \end{cases} \quad (3.14)$$

where the functions $\text{Randi}([a,b])$ and $\text{Uniform}([a,b])$ randomly produce an integer and real value from the uniform distribution of $[a, b]$, respectively. For simulating the case in which real optimal point change due to the disturbance, there is a quiescent period between the disturbances.

When several data points are generated by the proposed disturbance scenario, the real plant optimal points are affected as shown in Eq. (3.5-14) and the solution can be classified as presented in Figure 3.1 (a): Class 1 for disturbance cases 2 and 7; Class 2 for disturbance cases 3, 6 and 9; Class 3 for disturbance cases 1 and 4; Class 4 for disturbance cases 5 and 8. When the real optimum is located in Class 3 by the disturbance case 1 or 4, optimization performances of different initial point of manipulated variable are compared in Figure 3.1 (b-d). In each simulation, 10 iterations of modifier adaptation are performed. Even if the case of starting from Class 3 cannot reach the real optimum in 10 iterations, it keeps staying near the real plant optimum compared to the other cases. As a result, the cumulative error

between the real optimum and suboptimum of starting in Class 3 is smaller than those of starting in Classes 1, 2, 4 as shown in Figure 3.1 (d). This indicates that proper assignment of the initial point is important. The following sections describe how to use historical data and machine learning techniques to estimate good initial values in a feedforward manner in accordance with the disturbance changes.

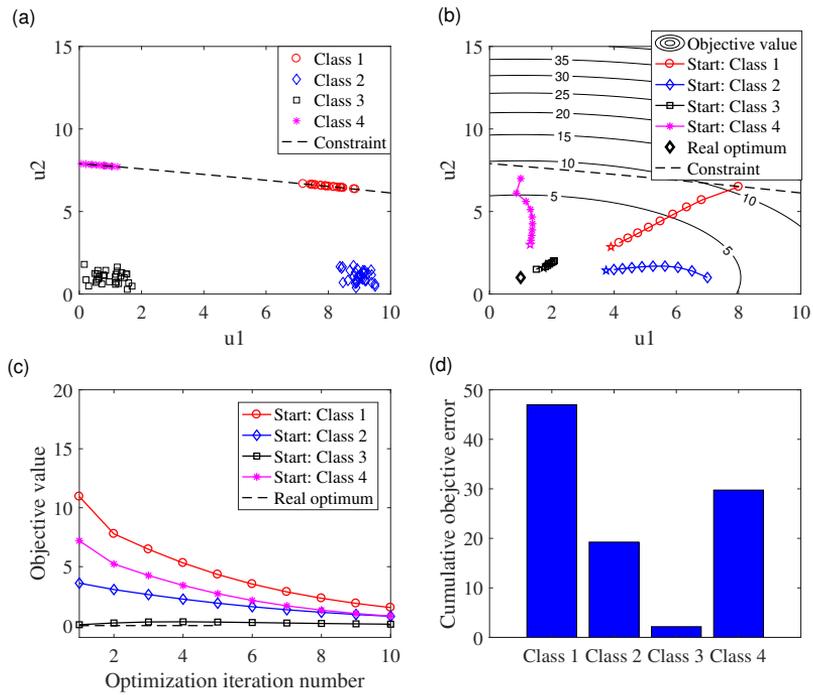


Figure 3.1: Optimal point change by the disturbance and resulting iterative optimization performances in the scheme of modifier adaptation: (a) 4 different classes of optimal point; (b) Iterative optimization result according to the starting points; (c) Objective values by iterative optimization according to the starting points; (d) Cumulative error between real optimum and sub-optimum by iterative modifier adapting optimization according to the starting points.

Table 3.1: Nominal parameter values for numerical example in issue 1

Par	Value	Par	Value	Par	Value	Par	Value
a_1	0.1	a_2	0.2	a_3	1	a_4	0.4
a_5	25	a_1^m	0.2	a_2^m	0.3	a_3^m	2
a_4^m	2	a_5^m	20	u_1^{m*}	5	u_2^{m*}	5
lb_1	0	lb_2	0	ub_1	10	ub_2	10

Machine learning generally refers to constructing a function that can explain correlations among the data. The neuron, the basic element of the function, is a linear sum of inputs into a nonlinear transfer function and can be expressed as:

$$y = f_{tf}(Ax + b) \quad (3.15)$$

where x , y and f_{tf} are the input, output and nonlinear transfer function, respectively. Representative nonlinear transfer functions for classification are sigmoid and softmax functions:

(Sigmoid)

$$y = \frac{1}{1 + e^{-z}} \quad (3.16)$$

(Softmax)

$$y_j = \frac{e^{z_j}}{\sum_{i=1}^K e^{z_i}}, \quad j = 1, \dots, K \quad (3.17)$$

where z and K are the linear sum of inputs and number of classes, respectively. A set of weights (A and b) that best describes the given data for whole neurons should be obtained. Especially, finding weights on machines with multilayer structure is called deep learning. In the deep structured machine, the output of the neuron serves as the input for the next layer's neurons. Early methods of deep learning such as deep belief network (DBN) apply unsupervised learning and find weights of each layer successively. Then these weights from unsupervised learning of each layer are set as the initial values for finding the whole weights in the structure by the supervised learning. Since the emergence of DBN, deep machine learning techniques have evolved

a number of times and several related techniques have been applied to improve the performance.

There have been three remarkable improvements in deep learning field to overcome the limitations and to enhance the predictive performance. First, nonlinear transfer function of neurons was replaced from the sigmoid to the rectifier linear unit or ReLU function defined as follows:

$$y = \max(0, z). \quad (3.18)$$

where z is the linear summation of input. When learning a deep structure by back propagation with sigmoid function bounded between 0 and 1, there occurs a problem of vanishing gradient, leading to poor weight estimates. The vanishing gradient problem of deep structure learning can be addressed by using the ReLU transfer function, which shows a linear behavior for positive values [27]. Second, new methods are proposed to give initial values of weights in gradient descent learning algorithm [20, 19]. For example, Xavier initialization has the following statistics instead of giving random initial weights:

$$\begin{aligned} \text{mean}(W) &= 0 \\ \text{Var}(W) &= \frac{2}{n_{in} + n_{out}} \end{aligned} \quad (3.19)$$

where W , n_{in} and n_{out} are the initial weights and the number of neurons feeding and being fed, respectively. By setting a random distribution of the initial weights, the signal can be kept in a reasonable range of values through many hidden layers [20]. Third, in order to deal with over-fitting problem given a limited number of training data and sampling noise, a dropout technique is proposed. The key idea

is to randomly select the units from the neural network to drop out and learn the weights with the remainder only. In actual applications, all the learned units are involved together. This prevents the units from co-adapting too much and reduces the overfitting compared with other regularization methods [17]. Due to the development of these various technologies, deep learning performance has improved and could be applied to commercialized technology in various fields.

The deep structured machines constructed for feedforward decision maker in the following sections commonly have 5 layers and 10 neurons in each layer, except a narrow structured case of the numerical example. From the first to the fourth layers, the nonlinear transfer functions are set as ReLU. For the last fifth layer, the nonlinear transfer functions are set as softmax. The decision maker finds the largest value among the results from the softmax transfer function and chooses it as the output. The weights of deep structured machine are initialized first by the Xavier's method and it provides a better learning performance compared to the conventional random initialization. Lastly, in order to avoid the over-fitting problem, the dropout strategy is applied. The dropout rate is set as 0.9 for each simulation case.

Based on the recent developments of deep learning techniques, a feedforward decision maker using historical data is proposed. The historical data for supervised machine learning involve the following input and output:

$$\begin{aligned} \text{Input} &:= D(i), \quad i = 1, \dots, I \\ \text{Output} &:= \text{Class}(i), \quad i = 1, \dots, I \end{aligned} \tag{3.20}$$

where D , $Class$ and I are the disturbance, suboptimal class of manipulated variable by unsupervised learning, and historical data index, respectively. The suboptimal manipulated variable for classification, $U^*(i)$, $i = 1, \dots, I$ is defined as:

$$U^*(i) := \arg \min_u \Phi(u(n), D(i)), \quad n = 1, \dots, j(i) \quad (3.21)$$

where Φ , u , $j(i)$ are the objective function, manipulated variable and iteration index of optimization for the i^{th} disturbance case. After classification of the suboptimal manipulated variable, representative values of each class such as centroid or average are determined. After the feedforward decision maker estimates a class given the current disturbance, it suggests a corrected initial value for the manipulated variable associated with the disturbance.

For the robustness of the proposed feedforward decision maker to address the noise and multiple disturbances occurring at the same time, several adaptation steps are involved in the proposed method. First, decision making by the constructed machine is used only where the disturbance is in confidence level. Because the proposed method has softmax transfer functions in the last layer and the result is suggested as normalized probabilities of each class, the confidence level for determining whether the decision maker works is defined as follows:

$$\max f_{FF}(d) > \eta \quad (3.22)$$

where f_{FF} and η are the feedforward decision maker and threshold of confidence level, respectively. If the output from the feed-forward decision maker is lower than the threshold value, the suggested ap-

proach works the same as the conventional modifier adaptation instead of suggesting an unreliable initial point. Second, when the same disturbance is observed repeatedly, the algorithm sets the initial point as the last suboptimal one corresponding to the disturbance. In this way, suboptimal values in the historical data can be learned successively. Figure 3.2 shows the structure of the proposed algorithm.

-
1. $k=k+1$
 2. Choose $u(k)$ by following logics:
 - 2.1. $d(k) \neq d(k-1)$
 - if) $d(k) \in \mathbf{D}$: for i where $d(k)=D(i)$, $u(k) = U^*(i)$
 - else) i. $I = I + 1$, $D(I) = d(k)$, $j(I) = 1$
 - ii. Classification $\{U^*(i), i = 1, \dots, I-1\}$ and Labelling.
 - iii. Construct the feed-forward decision maker
(Input: $\{D(i), i = 1, \dots, I-1\}$, Output: Labels from ii.)
 - iv. if) $d(k)$ in confidence level: $u(k)$ by feed-forward decision maker from iii.
else) $u(k) = u(k-1)$
 - 2.2. $d(k) = d(k-1)$
 $u(k)$ from previous modifier adaptation scheme.
 $j(i) = j(i) + 1$.
 3. Observe the value and gradient of objective and constraint.
 4. Update $U^*(i)$ by $U^*(i) := \arg \min_u \Phi(u(n), d(i)), n = 1, \dots, j(i)$
 5. Update the modifier and solve the modifier adaptation optimization to choose the next step manipulated variable.
 6. Return to step 1.
-

Figure 3.2: Proposed algorithm which combines the conventional modifier adaptation scheme and feed-forward decision maker by historical disturbance data and machine learning

Figure 3.3 illustrates the structural differences between the proposed method and the conventional modifier adaptation. Whereas the conventional modifier adaptation scheme has only feedback optimization and does not have a means to handle the disturbance change, the proposed method has both of feedback and feedforward schemes for the iterative optimization. In addition, a better (sub)optimal solution is successively updated in the historical data and utilized for future decision making. In this way, as the historical data are accumulated, a better operational policy with improved optimization performance is obtained. In the following section, the proposed method is applied to numerical and bio-process optimization examples to demonstrate its utility and discuss limitations and challenges.

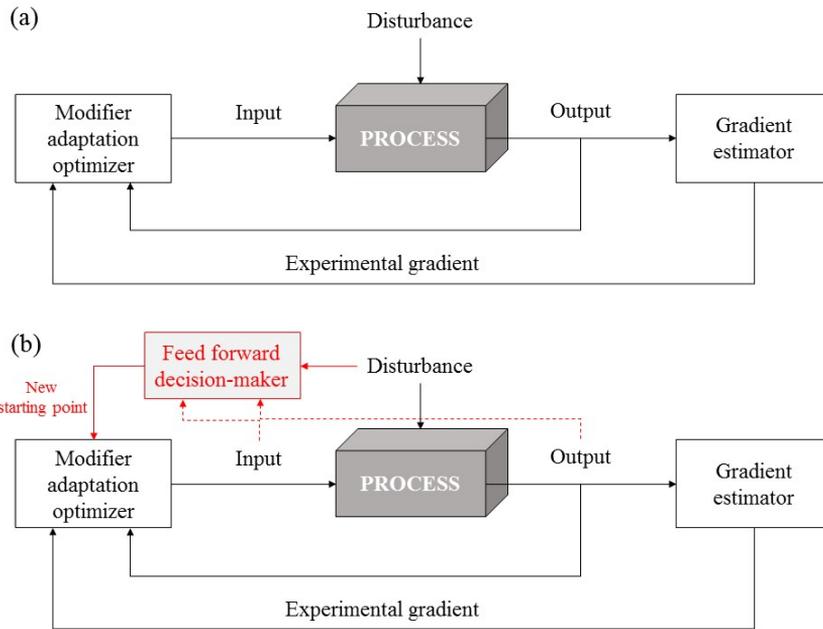


Figure 3.3: Compare of the algorithms: (a) Conventional modifier adaptation without feed-forward decision maker; (b) The proposed modifier adaptation with feed-forward decision maker.

3.1.2 Illustrative example

Plant and model equations of numerical example are the same as the one presented in Section 3.1.1. The following assumptions are made for the performance comparison between the proposed and conventional modifier adaptation methods.

1. Disturbances are irregularly generated four times in each 20 batches and large enough to change the real optimum.
2. All disturbances related to the changes of real plant optimum are measured.
3. Plant objective and constraint function values and their gradients are estimated correctly in each iteration.

We first consider the two cases: insufficient and sufficient amounts of historical data. The insufficiency of historical data occurs in the early and middle stages of the learning phase of the feedforward decision maker. In the early stage when very little historical data is available, there is no difference between the two methods since the feedforward decision maker cannot learn properly. Thus, the simulation results of illustrative examples deal with only the middle or transition stage with insufficient amount of historical data and the complete stage of learning that has seen sufficient amount of historical data.

Figure 3.4 shows the comparison of optimization performance for the ‘transition case’ ($k = 41, \dots, 60$) where the feedforward decision maker is still in the learning phase that has seen insufficient amount of historical data. Irregular changes in the disturbance occur four times at $k = 41, 46, 50$ and 55 . For simulating the transition

case, only the first to sixth disturbance cases are learned in advance using the historical data as shown in Figure 4 (a). When a disturbance occurs in the space learned in advance (the second disturbance at $k = 41$ and fourth disturbance at $k = 50$), the optimization performance by the proposed method outperforms the conventional modifier adaptation. However, when a disturbance occurs in the space not learned (the eighth disturbance at $k = 46$ and seventh disturbance at $k = 55$), the feedforward decision maker could not proceed and the optimization performance is not improved. Under the eighth disturbance at $k = 46$, the optimization performance of the proposed method is worse than that of the conventional modifier adaptation due to the incorrect estimation of initial point. For the seventh disturbance at $k = 55$, the optimization performance decreases because it is not within the confidence level and the feedforward scheme is not activated.

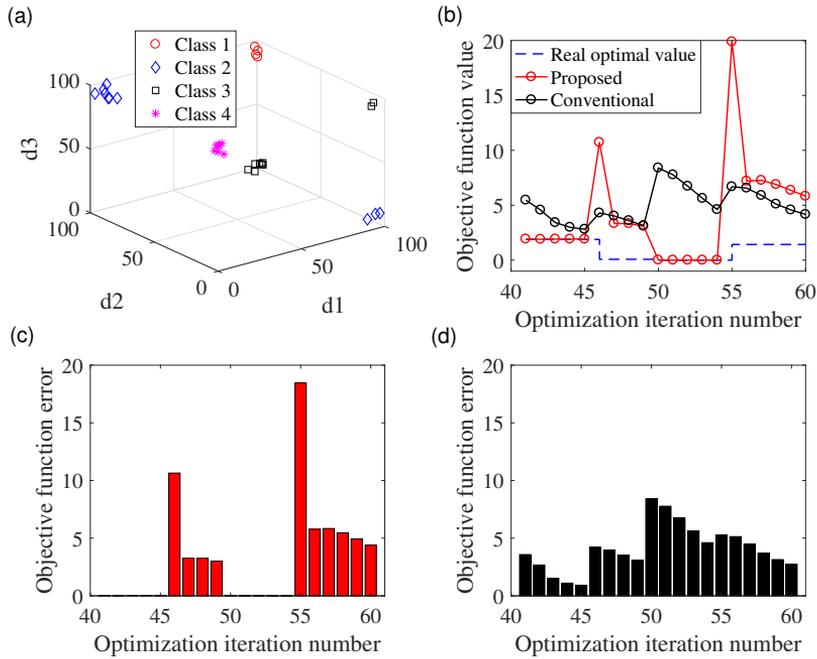


Figure 3.4: Iterative optimization performance in immature historical data situation: (a) Historical disturbance data set in which only from 1st to 6th disturbance cases have occurred; (b) Iterative optimization results by both of the proposed method and conventional modifier adaptation; (c) Objective function errors between real optimum and sub-optimum by the proposed method; (d) Objective function errors between real optimum and sub-optimum by the conventional modifier adaptation.

Figure 3.5 shows the results for the scenario of repeating disturbance. Irregular changes in the disturbance occur four times at $k = 41, 46, 50$ and 55 and only the first to sixth disturbance cases are learned in advance using historical data in order to reflect the transitional learning phase. At $k = 41$ and $k = 50$, a disturbance occurs in the space learned in advance and the feedforward decision maker works properly. On the other hand, the same disturbance not experienced is repeated at $k = 46$ and $k = 55$. Then the initial point at $k = 55$ is chosen as the best one for the disturbance at $k = 46$ by the step 2.1 in Figure 2. As a result, the optimization performance is improved at $k = 46$ compared with the conventional modifier adaptation as shown in Figure 5 (b). The subsequent optimization results are stored for updating the feedforward decision maker.

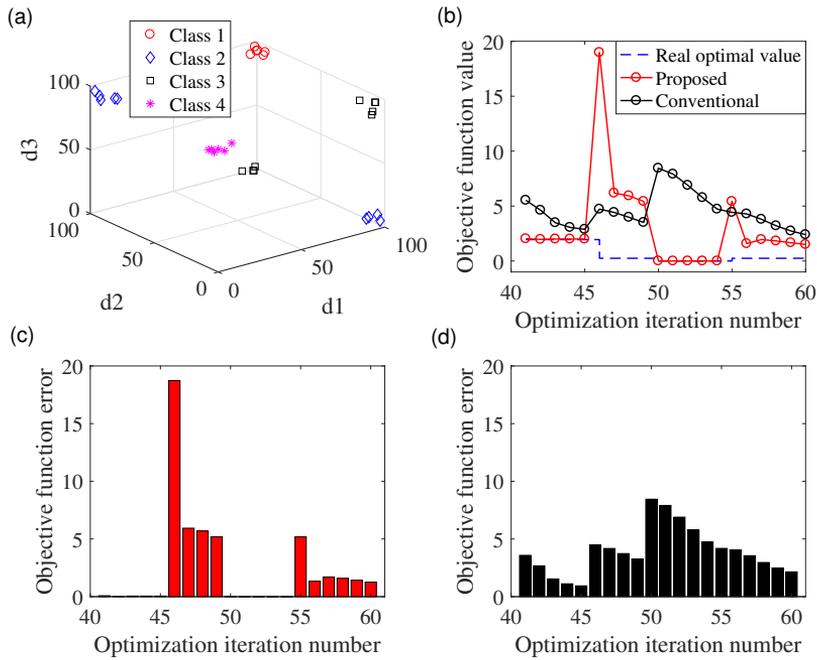


Figure 3.5: Iterative optimization performance when the same disturbance is repeated: (a) Historical disturbance data set in which only from 1st to 6th disturbance cases have occurred (immature historical data situation); (b) Iterative optimization results by both of the proposed method and conventional modifier adaptation when the same disturbance is repeated in $k = 46$ and $k = 55$; (c) Objective function errors between real optimum and sub-optimum by the proposed method; (d) Objective function errors between real optimum and sub-optimum by the conventional modifier adaptation.

Figure 3.6 shows the comparison of optimization performance for the case with sufficient historical data ($k = 81, \dots, 100$) where the learning has been completed. Irregular changes in the disturbance occur four times at $k = 81, 85, 91$ and 94 . In this case, all the disturbances (1-9) are learned sufficiently as shown in Figure 6 (a). Figure 6 (b)-(d) show that the proposed method outperforms the conventional modifier adaptation. The historical data for the machine learning consists of 100 data points of disturbance cases and related suboptimal manipulated variables. The feedforward decision maker has 5 layers consisting of 10 neurons of ReLU transfer function from the first to fourth layer and 10 neurons of softmax transfer function for the last fifth layer.

Figure 3.7 shows the simulation results for the case of sufficient amount of historical data, i.e., completed learning, with the feedforward decision maker having narrow structure. Irregular changes in the disturbance occur four times at $k = 81, 85, 91$ and 94 . The feedforward decision maker has a single layer consisting of 10 neurons with the softmax transfer function. At $k = 85$ and $k = 91$, the third and fourth disturbance cases are estimated well by the narrow structured feedforward decision maker. However, the second and fifth disturbance cases at $k = 81$ and $k = 94$ are not estimated properly by the narrow structured feedforward decision maker due to the nonlinear relationships including the exclusive or (XOR) relation between disturbance and class of real optimum as shown in Figure 7 (a). The narrow structured machine with only a single layer of softmax transfer function cannot learn the XOR logic, thus deep structured machine is needed to exploit the historical data having highly nonlinear relationships with the real optimum and estimate the class correctly. The

over-fitting problem stemming from the deep structure can be alleviated by the dropout strategy described in Section 3.2. The dropout rate was set 0.9 in all the simulation cases.

The above simulation results assume that the historical disturbances have a distribution of well separated class. However, it is also possible that the different disturbances occur at the same time and showing ‘overlapped distribution’ in time. For simulating this case, the following disturbance scenarios are considered:

Case = Randi([1, 8])

$$D(1) = \begin{cases} d_1 = \text{Uniform}([40, 100]) \\ d_2 = \text{Uniform}([0, 60]) \\ d_3 = \text{Uniform}([40, 100]) \\ k = [-0.013, -0.017, 0.004, 2.083, -0.004, 0.025, 0.004, 0.25] \end{cases} \quad (3.23)$$

$$D(2) = \begin{cases} d_1 = \text{Uniform}([40, 100]) \\ d_2 = \text{Uniform}([40, 100]) \\ d_3 = \text{Uniform}([40, 100]) \\ k = [0.008, 0.008, 0.017, 6.667, -0.004, 0.021, -0.008, 8.417] \end{cases} \quad (3.24)$$

$$D(3) = \begin{cases} d_1 = \text{Uniform}([40, 100]) \\ d_2 = \text{Uniform}([0, 60]) \\ d_3 = \text{Uniform}([0, 60]) \\ k = [0.004, -0.021, -0.008, 9.583, -0.008, -0.013, -0.013, 2.333] \end{cases} \quad (3.25)$$

$$D(4) = \begin{cases} d_1 = \text{Uniform}([40, 100]) \\ d_2 = \text{Uniform}([40, 100]) \\ d_3 = \text{Uniform}([0, 60]) \\ k = [0.008, 0.021, 0.004, -1.167, 0.004, -0.013, -0.017, 2.083] \end{cases} \quad (3.26)$$

$$D(5) = \begin{cases} d_1 = \text{Uniform}([0, 60]) \\ d_2 = \text{Uniform}([0, 60]) \\ d_3 = \text{Uniform}([40, 100]) \\ k = [0.008, -0.021, -0.004, 1.667, -0.013, 0.008, -0.013, 10] \end{cases} \quad (3.27)$$

$$D(6) = \begin{cases} d_1 = \text{Uniform}([0, 60]) \\ d_2 = \text{Uniform}([40, 100]) \\ d_3 = \text{Uniform}([40, 100]) \\ k = [0.008, -0.021, 0.004, 9.917, 0.013, -0.013, -0.008, 2.083] \end{cases} \quad (3.28)$$

$$D(7) = \begin{cases} d_1 = \text{Uniform}([0, 60]) \\ d_2 = \text{Uniform}([0, 60]) \\ d_3 = \text{Uniform}([0, 60]) \\ k = [0.013, 0.008, 0.013, 9.25, 0.008, -0.021, -0.004, 9.5] \end{cases} \quad (3.29)$$

$$D(8) = \begin{cases} d_1 = \text{Uniform}([0, 60]) \\ d_2 = \text{Uniform}([40, 100]) \\ d_3 = \text{Uniform}([0, 60]) \\ k = [0.004, 0.013, 0.017, -0.5, -0.008, 0.013, -0.013, 8.75] \end{cases} \quad (3.30)$$

The historical data by this scenario are presented in Figure 8 (a). In this case, even the deep structured feedforward decision maker with sufficient amount of historical data may estimate the initial point incorrectly. Irregular changes in disturbance occur four times at $k = 81, 85, 91$ and 94 . The optimization performances by the proposed method and the conventional modifier adaptation are compared in Figure 8 (b)-(d). Despite using the deep structure and sufficient amount of historical data, the initial point was chosen as the last manipulated variables like the conventional modifier adaptation due to its confidence level is lower than the threshold.

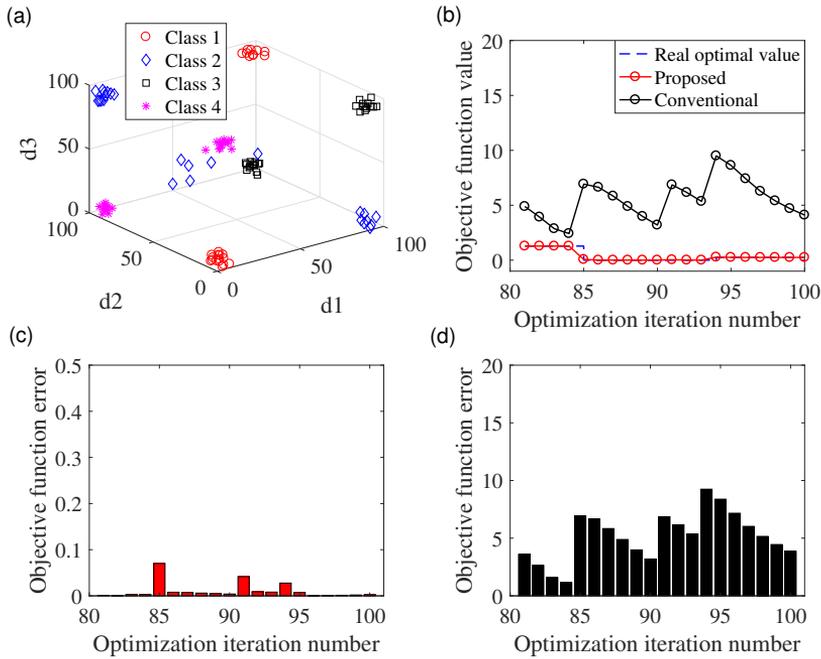


Figure 3.6: Iterative optimization performance in mature historical data situation: (a) Historical disturbance data set in which from 1st to 9th disturbance cases have sufficiently occurred; (b) Iterative optimization results by both of the proposed method and conventional modifier adaptation; (c) Objective function errors between real optimum and sub-optimum by the proposed method; (d) Objective function errors between real optimum and sub-optimum by the conventional modifier adaptation.

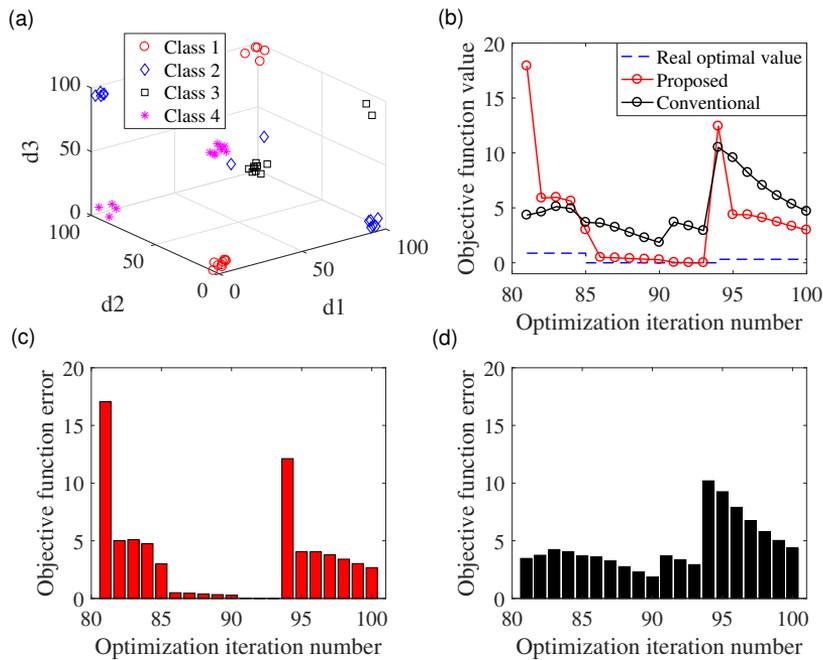


Figure 3.7: Iterative optimization performance by narrow structure feed-forward decision maker in mature historical data situation: (a) Historical disturbance data set in which from 1st to 9th disturbance cases have sufficiently occurred; (b) Iterative optimization results by both of the proposed method and conventional modifier adaptation; (c) Objective function errors between real optimum and sub-optimum by the proposed method having narrow structure feed-forward decision maker; (d) Objective function errors between real optimum and sub-optimum by the conventional modifier adaptation.

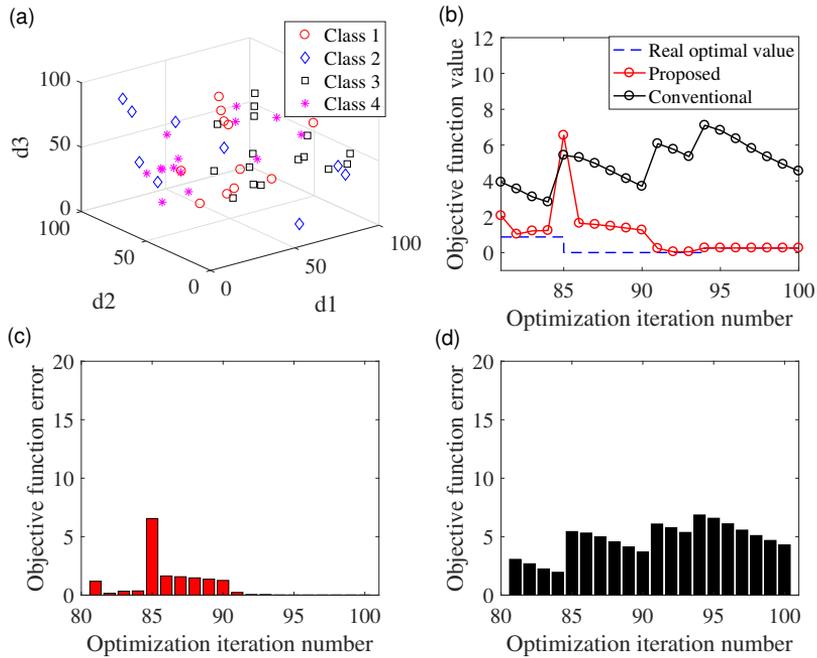


Figure 3.8: Iterative optimization performance in mature and complex historical data situation: (a) Historical disturbance data set in which from 1st to 9th disturbance cases have sufficiently occurred, but not had well-classified distribution; (b) Iterative optimization results by both of the proposed method and conventional modifier adaptation; (c) Objective function errors between real optimum and sub-optimum by the proposed method; (d) Objective function errors between real optimum and sub-optimum by the conventional modifier adaptation.

3.1.3 Run-to-run optimization of bioprocess

Consider run-to-run optimization of bioprocess with the following plant and model equations:

(Plant equation)

$$\begin{aligned}
 \min_F \quad & \Phi(F, T, pH, Pr_F, Pr_1, Pr_2) \\
 \text{s.t.} \quad & \Phi = Pr_F(F_1 + F_2 + F_3) - Pr_1P_1 - Pr_2P_2 \\
 & P_1 = k_{11}F_1^{1/2} + k_{21}F_2^{1/3} + k_{31}F_3^{1/2} \\
 & P_2 = k_{12}F_1^{1/3} + k_{22}F_2^{1/4} + k_{32}F_3^{1/2} \\
 & k_{11} = \frac{1}{1 + e^{-(T-25)}} + \frac{1}{1 + e^{-(pH-7)}} \\
 & k_{21} = \frac{e^{-(T-25)}}{1 + e^{-(T-25)}} + \frac{1}{1 + e^{-(pH-7)}} \\
 & k_{31} = \frac{1}{1 + e^{-(T-25)}} + \frac{e^{-(pH-7)}}{1 + e^{-(pH-7)}} \\
 & k_{12} = \frac{e^{-(T-25)}}{1 + e^{-(T-25)}} + \frac{e^{-(pH-7)}}{1 + e^{-(pH-7)}} \\
 & k_{22} = \frac{1}{1 + e^{-(T-25)}} + \frac{1}{1 + e^{-(pH-7)}} \\
 & k_{32} = \frac{e^{-(T-25)}}{1 + e^{-(T-25)}} + \frac{1}{1 + e^{-(pH-7)}} \\
 & P_1 \geq P_{1,min} \\
 & P_2 \geq P_{2,min} \\
 & F_1 + F_2 + F_3 \leq F_{max} \\
 & lb_1 \leq F_1 \leq ub_1 \\
 & lb_2 \leq F_2 \leq ub_2 \\
 & lb_3 \leq F_3 \leq ub_3
 \end{aligned} \tag{3.31}$$

(Model equation)

$$\begin{aligned}
& \min_F \Phi^m(F, Pr_F, Pr_1, Pr_2) \\
& \text{s.t. } \Phi^m = Pr_F(F_1 + F_2 + F_3) - Pr_1P_1^m - Pr_2P_2^m \\
& \quad P_1^m = k_{11}^m F_1^{1/2} + k_{21}^m F_2^{1/3} + k_{31}^m F_3^{1/2} \\
& \quad P_2^m = k_{12}^m F_1^{1/3} + k_{22}^m F_2^{1/4} + k_{32}^m F_3^{1/2} \\
& \quad P_1^m \geq P_{1,min} \\
& \quad P_2^m \geq P_{2,min} \\
& \quad F_1 + F_2 + F_3 \leq F_{max} \\
& \quad lb_1 \leq F_1 \leq ub_1 \\
& \quad lb_2 \leq F_2 \leq ub_2 \\
& \quad lb_3 \leq F_3 \leq ub_3
\end{aligned} \tag{3.32}$$

where the key states P_1 , P_2 , F_1 , F_2 and F_3 are the amounts of product 1 and 2, and feed 1, 2, and 3. Disturbances T , pH , Pr_F , Pr_1 and Pr_2 are the temperature, pH, prices of feed and product 1 and 2, respectively. The nominal parameter values for plant and model equations are presented in Table 3.2. It is assumed that the disturbances affecting the optimum of real plant are observable prior to each operation. The disturbance scenarios are given by:

$$\begin{aligned}
T &= \text{Uniform}([20, 30]) \\
pH &= \text{Uniform}([5, 9]) \\
[Pr_F, Pr_1, Pr_2] &= \text{Disturbance}(\text{Case}) \\
\text{Case} &= \text{Randi}([1, 8])
\end{aligned} \tag{3.33}$$

$$D(1) = \begin{cases} Pr_F = \text{Uniform}([15, 20]) \\ Pr_1 = \text{Uniform}([90, 100]) \\ Pr_2 = \text{Uniform}([50, 60]) \end{cases} \quad (3.34)$$

$$D(2) = \begin{cases} Pr_F = \text{Uniform}([15, 20]) \\ Pr_1 = \text{Uniform}([90, 100]) \\ Pr_2 = \text{Uniform}([90, 100]) \end{cases} \quad (3.35)$$

$$D(3) = \begin{cases} Pr_F = \text{Uniform}([5, 10]) \\ Pr_1 = \text{Uniform}([90, 100]) \\ Pr_2 = \text{Uniform}([50, 60]) \end{cases} \quad (3.36)$$

$$D(4) = \begin{cases} Pr_F = \text{Uniform}([5, 10]) \\ Pr_1 = \text{Uniform}([90, 100]) \\ Pr_2 = \text{Uniform}([90, 100]) \end{cases} \quad (3.37)$$

$$D(5) = \begin{cases} Pr_F = \text{Uniform}([15, 20]) \\ Pr_1 = \text{Uniform}([50, 60]) \\ Pr_2 = \text{Uniform}([50, 60]) \end{cases} \quad (3.38)$$

$$D(6) = \begin{cases} Pr_F = \text{Uniform}([15, 20]) \\ Pr_1 = \text{Uniform}([50, 60]) \\ Pr_2 = \text{Uniform}([90, 100]) \end{cases} \quad (3.39)$$

$$D(7) = \begin{cases} Pr_F = \text{Uniform}([5, 10]) \\ Pr_1 = \text{Uniform}([50, 60]) \\ Pr_2 = \text{Uniform}([50, 60]) \end{cases} \quad (3.40)$$

$$D(8) = \begin{cases} Pr_F = \text{Uniform}([5, 10]) \\ Pr_1 = \text{Uniform}([50, 60]) \\ Pr_2 = \text{Uniform}([90, 100]) \end{cases} \quad (3.41)$$

These scenarios involve both of the disturbance cases in Section 3.1.2: the temperature and pH are uniformly distributed; the prices of feed and product 1 and 2 are among the eight discrete cases.

For this example, only the simulation results with sufficient amount of historical data are presented. Figure 3.9 (a) shows the suboptimal values of the manipulated variables in the historical data. These data are classified by the unsupervised learning method and utilized to update the feedforward decision maker. The representative value of each class is the centroid and the optimal number of classes can change during the run-to-run optimization. Irregular changes in the disturbance occur at $k = 81, 85, 91$ and 94 . Figures 3.9 (b)-(d) show the comparison of optimization performance between the proposed and conventional modifier adaptation schemes. As the feedforward decision maker deals with the disturbance in advance, iteration can start close to the real optimum in all the cases. As a result, the optimization performance of the proposed method outperforms the conventional modifier adaptation. After the sufficient amount of data are accumulated, optimization performance is maintained by quickly responding to disturbances unless an unobserved disturbance is introduced.

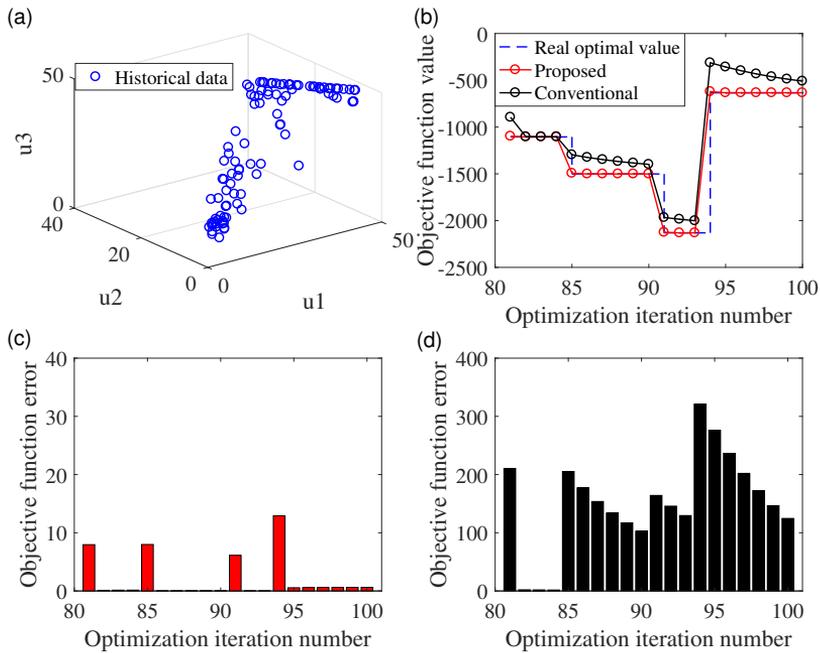


Figure 3.9: Iterative optimization performance in bio-process example: (a) Historical disturbance data set for constructing feed-forward decision maker; (b) Iterative optimization results by both of the proposed method and conventional modifier adaptation; (c) Objective function errors between real optimum and sub-optimum by the proposed method; (d) Objective function errors between real optimum and sub-optimum by the conventional modifier adaptation.

Table 3.2: Nominal parameter values for bio-process example in issue 1

Par	Value	Par	Value	Par	Value	Par	Value
$P_{1,min}$	5	$P_{2,min}$	5	F_{max}	100	k_{11}^m	1
k_{21}^m	1	k_{31}^m	1	k_{12}^m	1	k_{22}^m	1
k_{32}^m	1	lb_1	0	lb_2	0	lb_3	0
ub_1	50	ub_2	50	ub_3	50		

3.1.4 Concluding remarks

In order to deal with a large change of real optimum by disturbance, a new algorithm is proposed. It incorporates a feedforward decision maker, constructed by the historical data and machine learning technique, into the conventional modifier adaptation. The learned machine has a deep structure to cope with general case of disturbance distribution. Numerical and bioprocess examples are simulated to compare the performances of the proposed method and the conventional modifier adaptation. From the simulation results, it is concluded that the feedforward decision maker provides better starting points and improves the performance of the iterative optimization compared to the conventional modifier adaptation.

There may be several factors that degrade the optimization performance of the proposed method. The first is the problem of insufficient amount of data in the early phase of learning and overlapping of different disturbances in the historical data, which were addressed in the numerical example in Section 3.1.2. These issues can be addressed to some extent if a sufficient amount of data is accumulated. Second, there may exist a disturbance that may change the optimum but is not observed. If this disturbance is correlated with the measured ones, it can be indirectly reflected in the feedforward decision maker. However, if the unmeasured disturbance is completely independent one, the effect of this disturbance cannot be learned by the feedforward decision maker with the historical data. Third, the noise in the measured disturbance can degrade the performance of the feedforward decision maker. This problem can be alleviated by accumulation of more amounts of disturbance data and by data compression

technique such as principal component analysis (PCA) or partial least squares (PLS).

The proposed method can be applied to the optimization of dynamic process such as a fed-batch reactor in a limited sense. In this case, disturbances can occur in the middle of the process, which poses a challenging problem of estimating the value and gradient of the objective and constraint functions in real time. In addition, there may be an issue of computational time of updating the feedforward decision maker and solving the modified optimization problem. In addition, because disturbances can affect the system's dynamics differently at each time, the feed-forward decision maker should be designed adaptively in time, which limits the applicability of the proposed method. In conclusion, the proposed method is suitable for static or run-to-run optimization.

3.2 Issue 2: Experimental gradient estimation under noisy and multivariate condition

3.2.1 Importance of the gradient estimation for the modifier adaptation

A key element of the implicit and explicit approaches is estimating the gradients of the plant given input variables, referred to as experimental gradients [3]. Various methods have been proposed for appropriate gradient estimation. In the case of parametric mismatch, a model-based gradient estimation approach, called the neighboring extremal method, can be utilized [41]. A straightforward approach exists, which perturbs each input around the current operating point

and estimates the corresponding gradient [37, 43]. Roberts' algorithm using the finite difference is the most representative case [37]. However, these approaches are not practical because it requires that a perturbation should be performed for each input variable at every iteration [42]. Meanwhile, there are other gradient estimation methods that do not require input perturbations but instead use one or several past operating points. Brdyś and Tatjewski firstly proposed a finite difference scheme to estimate the gradient based on past operating points [39, 40]. Compared to the original ISOPE [37], this algorithm is called dual ISOPE because it has two conflicting objectives, the primal objective of improving the plant operation and the dual objective of estimating the experimental gradient [42]. In order to avoid the ill-conditioned or non-singularity problem, the dual ISOPE adds a constraint related to the condition number of updated input matrix.

Combining the advantages of model-based and model-free methods, regression models can be effectively used to optimize unconstrained processes [7, 28]. Camacho et al. [7] employed partial least squares regression to estimate experimental gradients with several input and output data points. The estimates are further used to update the operation strategy, e.g., feed rate, in a run-to-run fashion. Compared with model-free approaches, regression models achieve faster convergence to the plant optimum. However, regression models handle constraints after the optimization in an ad hoc manner.

This thesis extends the regression-based method proposed by Camacho et al. for unconstrained optimization to the constrained case by combining it with the modifier adaptation scheme. We evaluate the resulting method by simulating a highly multivariate system comprising run-to-run optimization of a fed-batch bioreactor having 50 ma-

nipulated variables, constraints, and model-plant mismatches. Compared to conventional methods of gradient estimation, the regression-based gradient estimation shows improved performance of optimization. Especially, several latent variable space model-based approaches, such as the partial least squares and principal component analysis, outperform the other ones in the case of optimizing the highly multivariate systems. The effects of the number of principal components and data points for gradient estimation on the optimization performance are studied under several simulations when using the latent variable space-based approaches. Based on these results, a moving average input update strategy and an algorithm that satisfies both fast convergence and stability near the KKT point are proposed.

3.2.2 Motivational example: Run-to-run optimization of bioreactor

The example system to be optimized is a fed-batch bioreactor with constraints related to the final values of key variables and manipulated variables. The manipulated variables to be decided for optimization are the step-wise 50 input feed rates which are discretized by time along the operation. The system and an arbitrary simulation results are illustrated in Figure 3.10. The optimization problem with plant equations is:

$$\begin{aligned}
& \max_{u_1, \dots, u_{50}} w_1 P(t_f) - w_2 \sum_{i=1}^{50} u_i \\
& \text{s.t. } X(t_i) = X(t_{i-1}) + \int_{t_{i-1}}^{t_i} ((\mu(t) - \mu_d) X(t) - D(t)X(t)) dt \\
& S(t_i) = S(t_{i-1}) + \int_{t_{i-1}}^{t_i} \left(- \left(\rho - \frac{\mu(t)}{Y_{XS}} + \frac{\pi(t)}{Y_{PS}} \right) X(t) + D(t)(S_{in} - S(t)) \right) dt \\
& P(t_i) = P(t_{i-1}) + \int_{t_{i-1}}^{t_i} (\pi(t)X(t) - D(t)P(t)) dt \\
& V(t_i) = V(t_{i-1}) + \int_{t_{i-1}}^{t_i} u(t) dt \\
& D(t) = \frac{u(t)}{V(t)} \\
& \mu(t) = \frac{\mu_m S(t)}{K_S + S(t)} \\
& \mu_d(t) = \frac{\mu_{dm} P(t)}{K_P + P(t)} \\
& \pi(t) = \alpha \mu(t) \\
& [X(t_0), S(t_0), P(t_0), V(t_0)] = [X_{ini}, S_{ini}, P_{ini}, V_{ini}] \\
& V(t_f) \leq \bar{V} \\
& S(t_f) \leq \bar{S} \\
& lb \leq u_i \leq ub \\
& t_i = t_{i-1} + (t_f - t_0)/50 \\
& u(t_{i-1} \leq t < t_i) = u_i \\
& i = 1, \dots, 50
\end{aligned} \tag{3.42}$$

where the variables X , S , P , S_{in} , V , u , D , μ , μ_d and π are the concentrations of biomass, substrate, product and feeding source, work-

ing volume, 50 discretized input feed rate, dilution rate, growth and death rate and production rate, respectively. The parameters t_0 , t_f , X_{ini} , S_{ini} , P_{ini} , V_{ini} , μ_m , μ_{dm} , K_S , K_P , lb , ub , \bar{V} , \bar{S} , w_1 and w_2 are the time to start and finish the operation, initial values of X , S , P and V , maximum growth and death rate, Michaelis constants related to S and P , lower and upper bounds of input feed rate, upper bounds of V and S at the final time and coefficients of objective function, respectively. The nominal values of the parameters are presented in Table 3.3.

For simulating the case of model-plant mismatch, the optimization by model equations is defined:

$$\begin{aligned}
 & \max_{u_1, \dots, u_{50}} \sum_{i=1}^{50} - (u_i - \bar{u})^2 \\
 & \text{s.t.} \sum_{i=1}^{50} u_i \leq \bar{V}_m \\
 & \sum_{i=1}^{50} u_i^2 \leq \bar{S}_m \\
 & lb \leq u_i \leq ub
 \end{aligned} \tag{3.43}$$

where \bar{u} , \bar{V}_m and \bar{S}_m are coefficients for model and their values are also remarked in Table 3.3. The optimization of this highly multivariate system is conducted in run-to-run way by modifier adaptation explained in the following section.

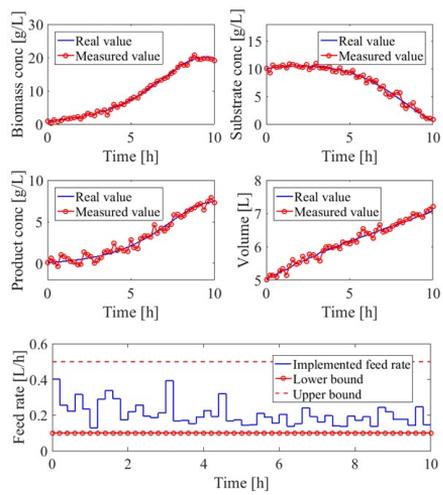
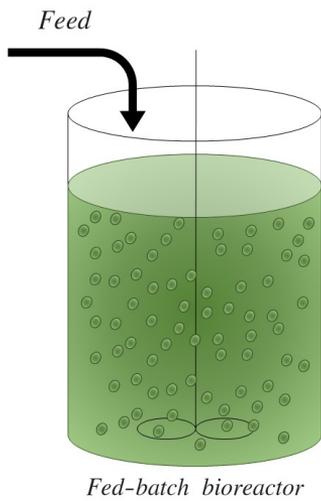


Figure 3.10: The fed-batch bioreactor system to be simulated for the comparison of gradient estimating methods. The key variables are the concentrations of biomass, substrate and product, and the working volume of the reactor. The manipulated variables are 50 discretized feed flow rates in step form.

Table 3.3: Nominal values for plant and model equations in issue 2

Parameter	Value	[Unit]	Parameter	Value	[Unit]
t_0	0	[h]	lb	0.1	[L/h]
t_f	10	[h]	ub	0.5	[L/h]
X_{ini}	1	[g/L]	\bar{V}	10	[L]
S_{ini}	10	[g/L]	\bar{S}	2	[g/L]
P_{ini}	0.1	[g/L]	w_1	10	[L/g]
V_{ini}	5	[g/L]	w_2	2	[h/L]
μ_m	0.9	[1/h]	\bar{u}	0.3	[L/h]
μ_{dm}	0.2	[1/h]	\bar{V}_m	6.5	[L]
K_S	10	[g/L]	\bar{S}_m	0.5	[g/L]
K_P	10	[g/L]			

3.2.3 Conventional experimental gradient estimation

The key element of the modifier adaptation scheme is the estimation of the gradient of the plant objective and constraint equations. It is worth noting that measurement noise has a significant effect on not only experimental gradient estimation, but also on the zeroth- and first-order modifiers. As a result, it degrades the performance of modifier adaptation. Conventional estimation schemes can be grouped into two categories according to whether set-point perturbation is utilized [6]. Although a model-based experimental gradient approach exists, referred to as the neighboring extremal method, we do not consider the method because it is only applicable parametric uncertainty or weak structural uncertainty. This work addresses estimation methods that can handle more general cases of model-plant mismatch, including strong structural uncertainty.

The finite difference approach is probably the most straightforward approach for estimating experimental gradients, and it was used by early versions of the ISOPE technique. It requires small perturbations of inputs at a current set point for calculating derivatives. In the forward finite difference approach, the experimental gradient is estimated as

$$\nabla_{ij}F(u(k)) := \frac{\partial F_i}{\partial u_j|_{u(k)}} = \frac{F_i(u(k) + he_j) - F_i(u(k))}{h} \quad (3.44)$$

where $\nabla_{ij}F : \mathbb{R}^{n_u} \rightarrow \mathbb{R}$, $F_i : \mathbb{R}^{n_u} \rightarrow \mathbb{R}$ (with $i = 1, \dots, n_G + 1$), $h \in \mathbb{R}$ and $e_j \in \mathbb{R}^{n_u}$ are the gradient of i^{th} output (objective or constraint functions) by j^{th} input, i^{th} output, step size and j^{th} unit vector. However, this approach is impractical since it requires n_u perturba-

tions at each iteration. Especially for optimization of steady state system, it needs to wait for a new steady state after each perturbation, which will take too long for systems with slow dynamics.

Alternatively, Broyden's formula is utilized for estimating experimental gradient [43]. The Broyden's formula uses two points of current and last data to update experimental gradients as

$$\begin{aligned}\nabla F(u(k)) &= \nabla F(u(k-1)) + \frac{[\Delta F(u(k)) - \nabla F(u(k-1))\Delta u(k)]\Delta u(k)^T}{\Delta u(k)^T \Delta u(k)} \\ \Delta F(u(k)) &:= F(u(k)) - F(u(k-1)) \\ \Delta u(k) &:= u(k) - u(k-1)\end{aligned}\tag{3.45}$$

where $\Delta F_i : \mathbb{R}^{n_u} \rightarrow \mathbb{R}$ (with $i = 1, \dots, n_G + 1$) and $\Delta u(k) : \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_u}$ are the differences in output and input between the current point in time k and the past point in time $k - 1$, respectively. Unlike the finite difference approach, no additional perturbations are needed in this approach. However, additional constraints are needed to avoid the ill-condition by limiting truncation and noise error [42].

Recently, an extended version of finite difference method proposed by Brdyś and Tatjewski is the most widely used for estimating experimental gradients with various types of modifier adaptation schemes [6, 4, 42]. It does not require set point perturbation and utilizes past data points to estimate experimental gradients. The method

is given by

$$\begin{aligned}
\nabla F(u(k)) &= U(k)^{-1} \Delta F(k) \\
\Delta U(k) &:= \left[u(k) - u(k-1), \dots, u(k) - u(k-n_u) \right]^T \\
\Delta F(k) &:= \left[F(u(k)) - F(u(k-1)), \dots, F(u(k)) - F(u(k-n_u)) \right]^T
\end{aligned} \tag{3.46}$$

where $\Delta U(k) : \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_u \times n_u}$ and $\Delta F_i(k) \in \mathbb{R}^{n_F \times n_u}$ (with $i = 1, \dots, n_G + 1$) are the difference matrices in output and input between the current and past data points, respectively. If $\Delta U(k)$ is ill-conditioned, the gradient estimate becomes unreliable. Moreover, since the outputs are usually corrupted by measurement noise, it is recommended $U(k)$ be designed to be well-conditioned. For this, the following constraint on $\Delta U(k)$ is added

$$d(k) := \frac{\sigma_{\min}(\Delta U(k))}{\sigma_{\max}(\Delta U(k))} = \gamma^{-1}(\Delta U(k)) \geq \delta \tag{3.47}$$

where $d(k)$ and δ ($0 < \delta < 1$) denote the reciprocal of the condition number of $\Delta U(k)$ and threshold value. If $d(k)$ is too small, the errors in the measurements will be amplified and the gradient estimate will be corrupted by noise. Brdyś and Tatjewski reformulated the modified optimization problem to take into account future requirements of the gradient estimation [39]. Adding the above constraint introduces an additional non-convexity in the optimization, as the next operating point must not belong to the hyperplane formed with $u(k), \dots, u(k-n_u-1)$. This drawback can be relaxed by splitting the problem into two; finding the value of $u(k+1)^*$ in both sides of the hyperplane of the modifier adaptation problem and applying one of them to the

process [35].

Among the conventional gradient estimation methods, the Brdyś and Tatjewski's method is utilized for the optimization of the highly multivariate fed-batch bioreactor. The condition number related to the constraint described by Eq. (3.47) is reflected in post-processing through the algorithm proposed by Gao and Engell [6]. Namely, if $d(k)$ is smaller than the threshold δ , a new operation point is suggested and applied to the process to make the matrix $U(k)$ well-conditioned. For monitoring of the condition number of $U(k)$ along the iterative optimization, its value at each iteration is presented in Figure 3.11 (b). Although the condition number is controlled by Gao and Engell's algorithm, it cannot be converged to the real process optimum, as presented in Figure 3.11 (a). In repeated simulations, it generally fails to converge in the vicinity of the plant optimum. As the wrong estimation of the gradient is the key factor of the poor optimization performance, the gradient estimation method must be improved for modifier adaptation in highly multivariate systems.

3.2.4 Regression based gradient estimation and its application to modifier adaptation

The simplest linear regression method, multivariate linear regression (MLR) and its modified version, robust MLR (RMLR), can be utilized to estimate the experimental gradients. MLR, which is a special case of general linear regression, considers more than one input variables while restricting the number of output variables to one [45].

The formulation of MLR is as follow:

$$\begin{aligned}
 \widehat{F} &= UA^T \\
 A^* &= \mathop{\text{arg min}}_A (F - \widehat{F})^T (F - \widehat{F}) \\
 \nabla F(u(k)) &= A^*
 \end{aligned} \tag{3.48}$$

where $\widehat{F}_i \in \mathbb{R}^p$ (with $i = 1, \dots, n_G + 1$), $F_i \in \mathbb{R}^p$ (with $i = 1, \dots, n_G + 1$), $U \in \mathbb{R}^{p \times n_u}$, $A \in \mathbb{R}^{n_u}$, $A^* \in \mathbb{R}^{n_u}$ and p are the estimated and real values of output matrices (constraints or objective function value), input matrix, arbitrary and optimal coefficients for the MLR model and the number of data points, respectively. For input and output matrices, they are normalized first and then used for MLR and this is common to all subsequent regression methods. RMLR has the same formulation with MLR, but weighting factors are introduced to reduce the effect of outliers by the noise. For more information about RMLR, refer to [44]. The experimental gradients are estimated as the parameter A^* around the average of data points [7, 28]. Though the plant equations are generally nonlinear, the gradient can be approximated well when the region of data set is small enough to ignore the nonlinearity.

MLR is utilized to estimate the experimental gradient of the highly multivariate system and combined with the modifier adaptation scheme. Simulation results from MLR are presented in Figure 3.12, and no increase of optimization performance is observed compared to Brdyś and Tatjewski's method in Figures 3.11. The condition number is controlled to a similar level when compared to the conventional method, as presented in Figures 3.11(b) and 3.12(b).

In order to increase the optimization performance of the MLR-

based method, a moving average input update is proposed. The motivation is that the best fit of the estimated gradient from the data is expected to be at the center (or averaged point) of where the data are distributed because the proposed method uses numerous data points for regression. With similar intent, the method by Gao et al. [46, 47] has a step of selecting the covariance-based search space to define a trust region for regression, and the algorithm by Camacho et al. [7, 28] has a similar moving average step. The difference between the proposed method and Camacho's algorithm is that it involves an exponential filtering term to increase the robustness of the iterative updating procedure, similar to conventional modifier adaptation schemes [5]. The proposed moving average update rule with exponential filter is:

$$\begin{aligned}
u^{*,imp}(k+1) &= \frac{(I - \alpha)}{N} (u^{*,imp}(k - N + 1) + \dots + u^{*,imp}(k)) \\
&\quad + \alpha u^*(k+1) \\
&:= (I - \alpha) \bar{u}^{*,imp}(k) + \alpha u^*(k+1)
\end{aligned} \tag{3.49}$$

where $u^{*,imp}(k+1)$, $\bar{u}^{*,imp}$, $u^*(k+1)$ and α are the input for implementation at time $k+1$, average of implemented inputs in the past N time steps for estimating gradients, computed new input at time $k+1$ by modifier adaptation in Eq. (2.1), and the exponential filter gain matrix for the moving average input update strategy, respectively. When the moving window input update strategy is applied, the exponential filter for the modifiers in Eq. (2.2) can be disabled as they have similar roles of suppressing the input updates. It can be proven that the converged input from Eq. (3.49) satisfies the KKT conditions of plant equations.

Theorem 3.1. *Let the user be able to arbitrarily change the values of exponential filter gain and the modifier adaptation scheme of Eq. (2.2), and let the moving average input update strategy be converged. If no measurement noise exists and model adequacy is satisfied, a converged solution of Eq. (3.49) or the KKT points produced by the moving average input update strategy satisfy the KKT conditions of the plant equations.*

Proof If there is no exponential filter for the modifiers and $N = 1$ in Eq. (3.49), it is a variant of filtering techniques proposed by Marchetti et al. [5]. In that paper, it was proven that the converged KKT point of the modifier adaptation scheme satisfied the plant KKT conditions when the input exponential filter was used and the following conditions are satisfied: (1) the exponential filter gain is nonsingular, (2) the modifier adaptation scheme in Eq. (2.2) is converged, and (3) the filtered input is converged. For more general formulation having N data points, the input exponential filter becomes

$$u^{*,imp}(k+1) = \sum_{i=k-N+1}^k \gamma_{i-k+N} u(i) + \gamma_{N+1} u^*(k+1) \quad (3.50)$$

$$\sum_{i=1}^{N+1} \gamma_i = I$$

where γ is arbitrary matrix for filtering. The moving average input update in Eq. (3.49) can be seen as a special case of the above general input filter by Marchetti et al. [5]. The exponential filter gain can be set to be nonsingular by the user, and the other conditions are satisfied by the assumption. Thus, the converged solution produced

by modifier adaptation with the moving average input update is also a KKT point of the plant equations when the same conditions are satisfied. ■

The moving average input update is applied to the MLR-based modifier adaptation scheme, and the simulated results are presented in Figure 3.13. Compared to Figure 3.12 (a), the proposed method shows improved convergence performance with reduced fluctuation magnitude, as shown in Figure 3.13 (a). This property occurs because the proposed input updating rule suppresses excessive changes of the manipulated variables. However, it increases the frequency of the fluctuation along the iterative optimization, as shown in Figure 3.13 (a).

As known in the basics of regression, the coefficients estimated from normally distributed data also have normal distributions and they are uncorrelated each other. By using the concept of standard error of the mean, when the data has a normal distribution, its estimated gradient also shows a normal distribution and its standard deviation is s/\sqrt{N} , where s is the sample-based estimation of the standard deviation of the population and N is the number of data points. That is, more data points bring more robust estimation of the gradient. The conventional Brdyś and Tatjewski's method uses square input matrix and its inversion to calculate the gradient as in Eq. (3.46). However, it can be modified to use more data points and the pseudo-inverse matrix to increase the robustness of gradient estimation.

Unfortunately, an issue exists regarding the convergence rate. To avoid under-determined or infinite solutions, the number of starting perturbations used to construct the MLR model must be larger than the number of variables. However, in the case of optimizing the

highly multivariate system, this required number of starting perturbations is too large to implement in the optimization of real processes. Thus, when optimizing real processes having a large number of manipulated variables, the scarcity of data points is inevitable. Simulation results for different numbers of data points are presented in Figure 3.14. The cases are divided into categories, where Nw is the number of data points utilized for gradient estimation by MLR as well as the size of moving average window: under-determined cases ($Nw = 10$ and $Nw = 30$), the square matrix case ($Nw = 50$), and an over-determined case ($Nw = 70$). In the cases having an insufficient number of data points (i.e., under-determined), the gradient of the major variable having the largest column norm is determined in advance and this procedure is repeated until the number of obtained gradients equals the rank of the input matrix. The remaining gradients related to the minor variables are set to zero. As shown in Figure 3.14 (a), although the solutions based on insufficient data points and MLR with the moving average update strategy cannot approach the KKT point perfectly, they show similar convergence tendency compared to the other cases. The reason that they do not reach the exact KKT point is the lack of gradient information for the minor variables. As expected, when more data points are used for modeling, the performance of regression-based modifier adaptation is improved, as shown in the diamond line of Figure 3.14 (b). Compared to the other cases, the over-determined case shows the least fluctuation of the objective function value along the iterative input updates.

On the other hand, the center for modifier update can be changed in the manner of moving average for the conservative update and robustness. That is, the modifier adaptation scheme in Eq. (2.1) can be

modified as following equations:

$$\begin{aligned}
u^*(k+1) &:= \arg \min_u \Phi_m(u) := \Phi(u) + \delta^{\Phi,imp}(\bar{u}(k)) + \lambda^{\Phi,imp}(\bar{u}(k))(u - \bar{u}(k)) \\
\text{s.t. } G_m(u) &:= G(u) + \delta^{G,imp}(\bar{u}(k)) + \lambda^{G,imp}(\bar{u}(k))(u - \bar{u}(k)) \leq 0 \\
\bar{u}(k) &= \frac{u^*(k-N+1) + \dots + u^*(k)}{N}
\end{aligned} \tag{3.51}$$

where $\bar{u}(k)$ is the moving average point of the modifier update and N is the number of moving average window. However, there is a problem that the plant values of objective and constraint function at $\bar{u}(k)$ are unknown when calculating the zeroth-order modifier through this approach. Thus, it is proposed that the average of plant values ($\bar{\Phi}_p(k)$, $\bar{G}_p(k)$) substitute the values at $\bar{u}(k)$:

$$\begin{aligned}
\Phi_p(\bar{u}(k)) &\simeq \frac{1}{N} (\Phi_p(u(k-N+1)) + \dots + \Phi_p(u(k))) = \bar{\Phi}_p(k) \\
G_p(\bar{u}(k)) &\simeq \frac{1}{N} (G_p(u(k-N+1)) + \dots + G_p(u(k))) = \bar{G}_p(k).
\end{aligned} \tag{3.52}$$

As mentioned in Chapter 2, the zeroth-order modifier for the objective (δ^Φ) does not affect to the satisfaction of KKT conditions and can be eliminated. However, the zeroth-order modifier for constraint (δ^G) affects the feasibility at the converged point and the optimality. Thus, it should be proven that the approximation for the zeroth-order modifier in Eq. (3.52) is valid.

Theorem 3.2. *When the objective and constraints functions are continuous and the modifier scheme in Eq. (3.51) with the approximation for the zeroth-order modifier in Eq. (3.52) is converged, the average*

of plant equations equal the approximated values in Eq. (3.52).

$$\begin{aligned}\lim_{k \rightarrow \infty} \Phi_p(\bar{u}(k)) &= \lim_{k \rightarrow \infty} \bar{\Phi}_p(k) \\ \lim_{k \rightarrow \infty} G_p(\bar{u}(k)) &= \lim_{k \rightarrow \infty} \bar{G}_p(k)\end{aligned}\tag{3.53}$$

Proof Because the modifier of objective function can be omitted in the modifier adaptation, let the focus on the constraints equation. When the assumptions in Theorem 3.2 are valid and the converged point is u_∞ , the right side of equation in Eq. (3.53) becomes:

$$\begin{aligned}\lim_{k \rightarrow \infty} \bar{G}_p(u(k)) &= \lim_{k \rightarrow \infty} \frac{1}{N} (G_p(u(k - N + 1)) + \cdots + G_p(u(k))) \\ &= \frac{1}{N} (\lim_{k \rightarrow \infty} G_p(u(k - N + 1)) + \cdots + \lim_{k \rightarrow \infty} G_p(u(k))) \\ &= \frac{1}{N} (G_p(u_\infty) + \cdots + G_p(u_\infty)) \\ &= G_p(u_\infty)\end{aligned}\tag{3.54}$$

On the other hand, the left side of equation in Eq. (3.53) becomes:

$$\begin{aligned}\lim_{k \rightarrow \infty} G_p(\bar{u}(k)) &= G_p(\lim_{k \rightarrow \infty} \bar{u}(k)) \\ &= G_p(\lim_{k \rightarrow \infty} (\frac{1}{N} (u(k - N + 1) + \cdots + u(k)))) \\ &= G_p(\frac{1}{N} (\lim_{k \rightarrow \infty} u(k - N + 1) + \cdots + \lim_{k \rightarrow \infty} u(k))) \\ &= G_p(\frac{1}{N} (u_\infty + \cdots + u_\infty)) \\ &= G_p(u_\infty)\end{aligned}\tag{3.55}$$

The first equalization is valid because of the assumptions (continuity of G_p and converged values of $u(k)$). Thus, the averaged plant equa-

tions are equal to the approximated values in Eq. (3.52). ■

Corollary 3.1. *When the assumptions in Theorem 3.2 are valid, the converged point satisfies the KKT conditions of plant.*

Proof By the result of 3.2, the converged values of approximated objective and constraint functions are equal to the original values at the average point $\lim_{k \rightarrow \infty} \bar{u}(k) = \frac{1}{N}(u_\infty + \dots + u_\infty) = u_\infty$. Thus, with the result of Theorem 2.1, the converged point satisfies the KKT conditions of plant. ■

With this moving average point rule for the modifier update, more robust optimization can be achieved under the noisy measurement condition.

Theorem 3.3. *When the measurement noise has a normal distribution, $\Phi_{p,i} \sim \mathcal{N}(0, \sigma_i^2)$, $i = 1, \dots, n_g$, the variance can be reduced to σ_i^2/N by the approximated moving average point rule.*

Proof The measurement of constraint functions with noise are uncorrelated near the converged value. When the noisy variables having the normal distribution are uncorrelated, the variance of averaged values are:

$$\begin{aligned}
 \text{Var}(\bar{\Phi}_{p,i}) &= \text{Var}\left(\frac{1}{N}(G_{p,i} + \dots + G_{p,i})\right) \\
 &= \frac{1}{N^2} \text{Var}(G_{p,i} + \dots + G_{p,i}) \\
 &= \frac{1}{N^2} (\text{Var}(G_{p,i}) + \dots + \text{Var}(G_{p,i})) \\
 &= \frac{\sigma_i^2}{N}, \quad i = 1, \dots, n_g
 \end{aligned} \tag{3.56}$$

■

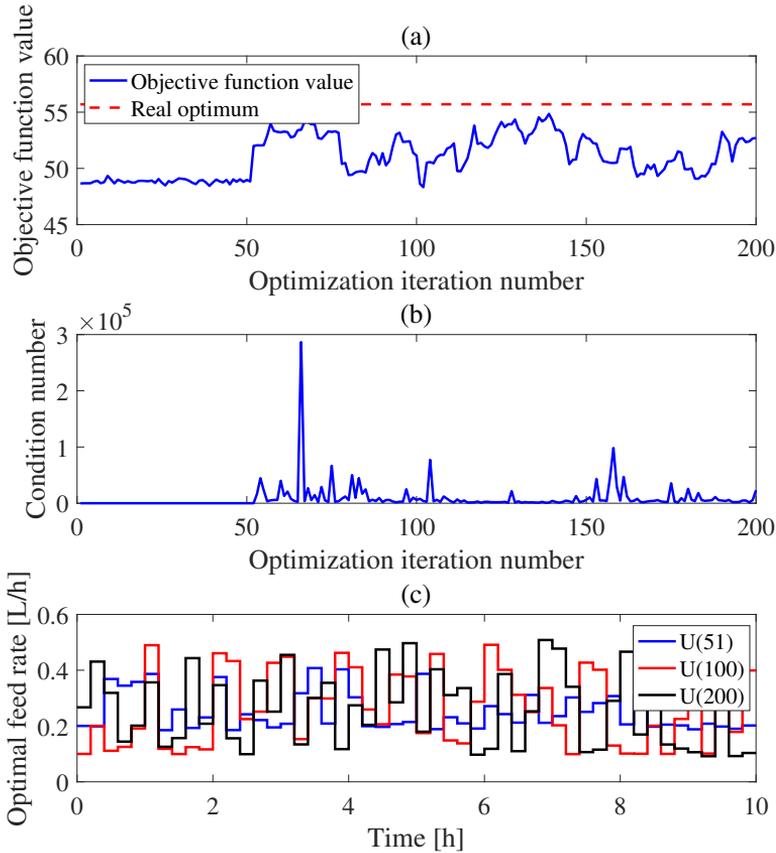


Figure 3.11: Simulation results by Brdyś and Tatjewski's gradient estimation method and modifier adaptation. It is assumed that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are $[50.1, 2.3]$ and $[52.1, 1.9]$, respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation.

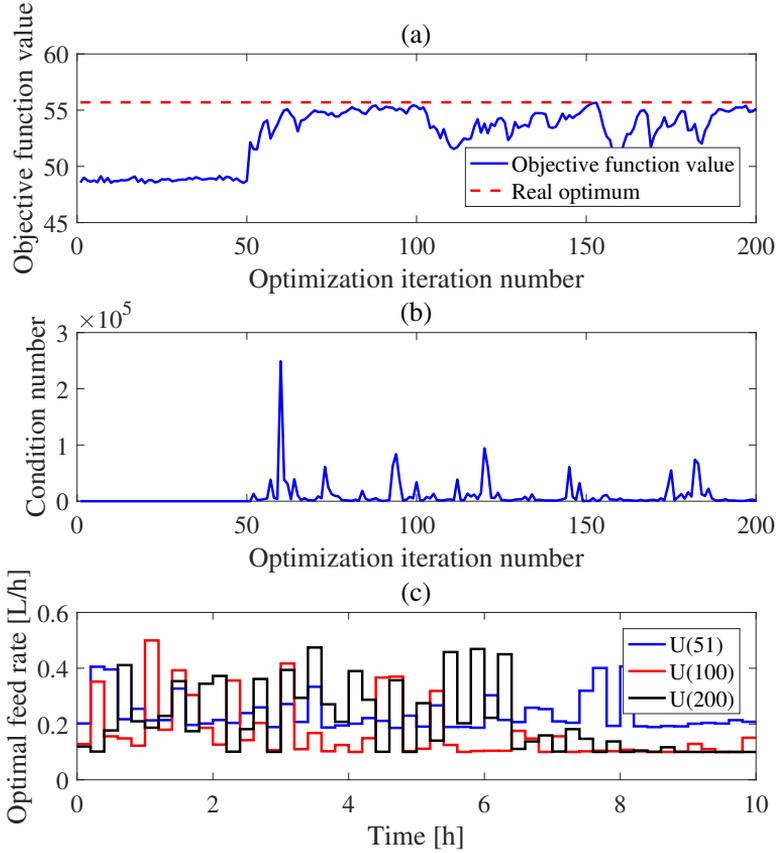


Figure 3.12: Simulation results by MLR based gradient estimation and modifier adaptation. It is set that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation and the number of data points for gradient estimation is 50. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are $[51.8, 2.5]$ and $[53.1, 1.5]$, respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation.

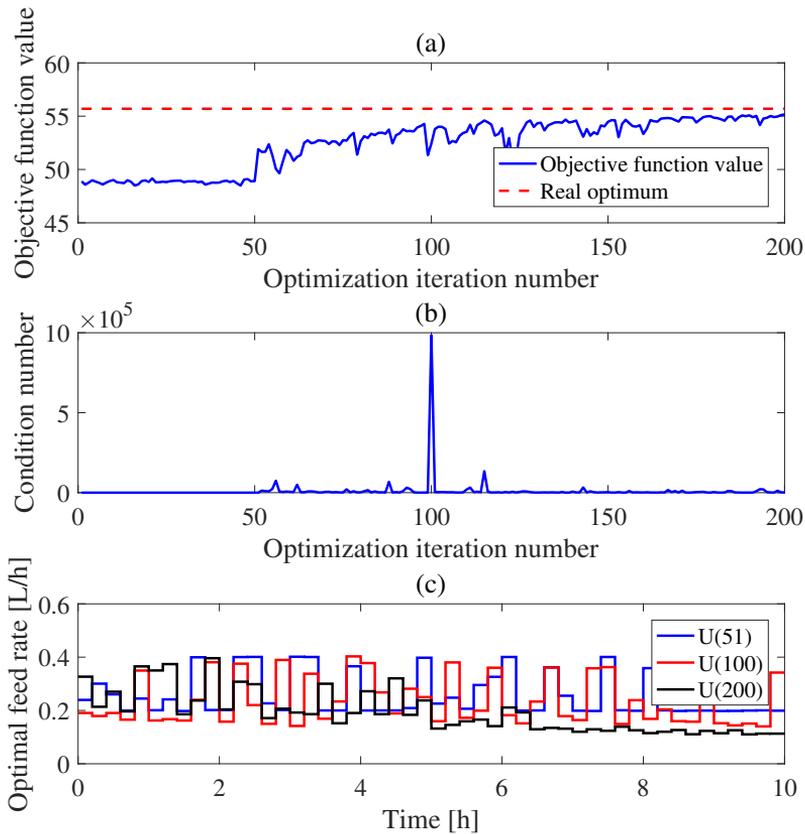


Figure 3.13: Simulation results by MLR based gradient estimation and modifier adaptation with the moving average input update strategy. It is set that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation, the number of data points for gradient estimation is 50 and the exponential filter gain for moving average input update is 0.8. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are $[52.8, 1.2]$ and $[54.7, 0.9]$, respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation.

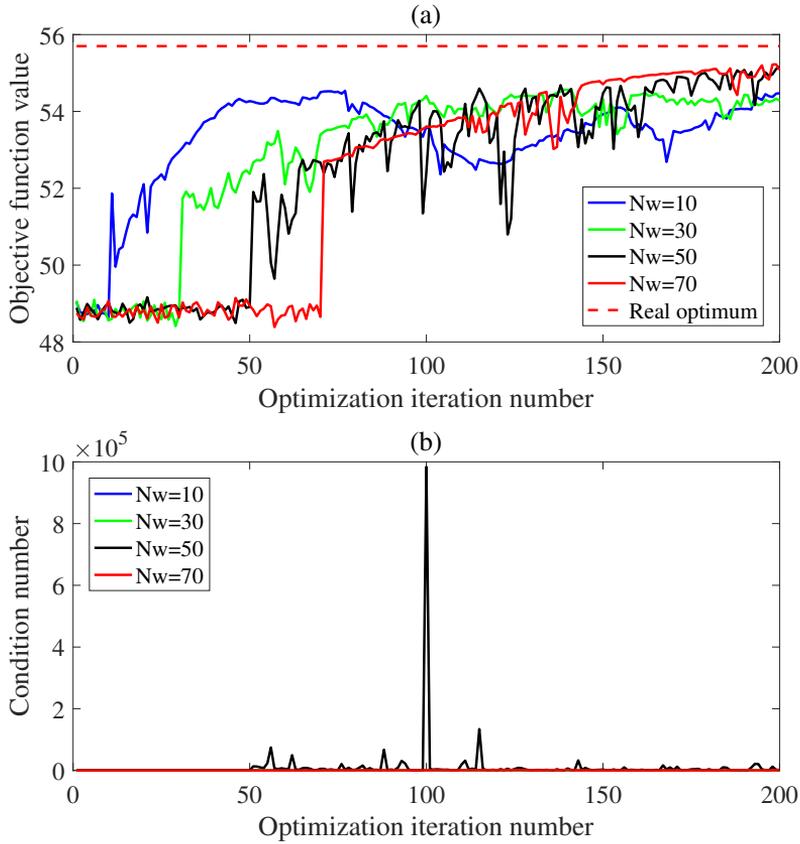


Figure 3.14: Simulation results by MLR with the moving average input update strategy depending on the number of data points for gradient estimation. (a) Optimization performance according to the number of data points, Nw . (b) The condition numbers of each data point case along the iterative optimization.

The experimental gradient can be estimated by modeling in latent variable space, beyond the general linear regression like MLR. Partial least squares regression (PLSR) is one of the most representative method to construct a linear model in the latent variable space [29]. The input and output data sets are projected to the latent variables as follows:

$$\begin{aligned}
U &= T_U P_U^T + E_U \\
\hat{U} &= T_U P_U^T \\
F &= T_F P_F^T + E_F \\
\hat{F} &= T_F P_F^T
\end{aligned} \tag{3.57}$$

where $U \in \mathbb{R}^{p \times n_u}$ and $F_i \in \mathbb{R}^p$ (with $i = 1, \dots, n_G + 1$) are the input and output matrices and hatted ones are their estimates. $T_U \in \mathbb{R}^{p \times N_p}$ and $T_F \in \mathbb{R}^{p \times N_p}$ are scores, $P_U \in \mathbb{R}^{n_u \times N_p}$ and $P_F \in \mathbb{R}^{n_G + 1 \times N_p}$ are loadings. $E_U \in \mathbb{R}^{p \times n_u}$ and $E_F \in \mathbb{R}^{p \times n_G + 1}$ are error matrices of input and output having negligible information that is not captured by the regression model. N_p is the number of latent variables or principal components. Then the least squares regression is carried out to identify the so-called inner relationship $T_F = T_U B$ in order to maximize covariance between the input and output score vectors:

$$\begin{aligned}
T_U &= U W (P_U^T W)^{-1} \\
\hat{F} &= U W (P_U^T W)^{-1} B P_F^T = U \Theta^T \\
\nabla F(u(k)) &= \Theta
\end{aligned} \tag{3.58}$$

where the matrices $\Theta \in \mathbb{R}^{n_G + 1 \times n_u}$, $B \in \mathbb{R}^{N_p \times N_p}$ and $W \in \mathbb{R}^{n_u \times N_p}$ are the PLSR model coefficients, inner relationship matrix and weighting matrix to ensure the input score's orthogonality [30]. As

PLSR estimates the experimental gradients in a latent variable space, the number of principal components can be adapted to suit the situation in each iteration, which is impossible for the MLR. Thus, it is expected that more precise estimation by PLSR is possible than by the MLR for highly correlated multi-variable system.

The simulation results from utilizing PLSR to estimate the experimental gradient of the highly multivariate system and combining it with the modifier adaptation scheme are presented in Figure 3.15. Fifty data points and four principal components are used to project to the latent variable. Compared to the optimization performance based on Brdyś and Tatjewski's method and MLR in previous sections, PLSR shows a largely improved convergence rate, as shown in Figure 3.15 (a). Although some condition numbers of the input matrix used in PLSR modeling show extremely high values in Figure 3.15 (b) compared to the previous cases, the effect of the ill-conditioned input matrix on the optimization performance is limited because the data points are projected to the latent variable space. The fact that no constraints are needed for the condition number of the input matrix is one of the key advantages of using PLSR with the modifier adaptation scheme to estimate the gradient. In addition, by projecting the data points to the latent variable space, the original data spaces are integrated together and measurement noise is filtered. As a result, it is expected that the effect of noise on the gradient estimation is largely reduced.

Even modifier adaptation using regression sometimes yielded a bad update direction because of noise and nonlinearity. However, their influence on gradient estimation is offset by the regression method, resulting in more robust convergence than by Brdyś and Tatjewski's

conventional method. It cannot achieve perfect convergence to the plant optimum, but it generally reaches the vicinity of the plant optimum in several simulations.

The moving average input update strategy is also applied to the PLSR-based modifier adaptation scheme, and simulation results are presented in Figure 3.16. Unlike the previous simulation results based on MLR, the utilization of the moving average input update does not increase the optimization performance in this case, as shown in Figure 3.16 (a). This strategy results in a slower convergence rate to the KKT point as compared to Figure 3.15 (a). Because the moving average input update suppresses excessive input updates and stabilizes the convergence, it is presumed to prevent PLSR from operating normally. However, if the nonlinearity of the system were more severe or the local optimal points were closer to each other, the combination of the moving average input update strategy and the PLSR-based modifier adaptation scheme would be effective.

As the number of manipulated variables is reduced to the number of principal components, the gradient can be estimated effectively with fewer data points than the MLR case. In order to show the efficiency of PLSR requiring fewer data points, simulations are conducted in 3 cases with 10, 30, and 50 data points (Nw). The optimization performance is compared in Figure 3.17. Compared to the MLR cases in Figure 3.17 (a), PLSR-based modifier adaptations with an insufficient number of data points show improved optimization performance, as shown in Figure 3.17 (a) and (b) where $Nw = 10$ and $Nw = 30$, respectively. Thus, PLSR seems more suitable than MLR for implementation in real process optimization having a large number of manipulated variables because it does not require a large num-

ber of starting perturbations. Comparing the three different cases in Figure 8 shows that the fluctuation magnitude increases as the number of data points decreases. Therefore, there exists a trade-off between the rate and stability of convergence to the KKT point. Given that the stopping criterion of the modifier adaptation is usually the gap between the present and previous objective function values for the manipulated variables, the fluctuation magnitude should be minimized to improve the performance of the optimization scheme. A method of adapting both the rate and stability of convergence is discussed in the following section.

In addition to the PLSR, the gradient can be estimated using another latent variable space modeling technique. Principal component analysis (PCA) is performed by projecting an arbitrary matrix $X \in \mathbb{R}^{p \times q}$ into latent variable space to maximize the variances of the components in X [48]:

$$\begin{aligned} X &= T_X P_X^T + E_X \\ \hat{X} &= T_X P_X^T \end{aligned} \tag{3.59}$$

where $T_X \in \mathbb{R}^{p \times N_p}$, $P_X \in \mathbb{R}^{q \times N_p}$, $E_X \in \mathbb{R}^{p \times q}$ are the score, loading and error matrices for the X , respectively. N_p ($\ll q$) is the number of principal components of the process optimally chosen according to the data. As the case of PLSR, the loading matrix acts as a new axis in the reduced dimensional latent variable space, and the score values represent the dynamics of each batch relative to all other ones.

A way to estimate the gradient using PCA is to apply PCA to the input matrix U first, then to perform the least square regression on the PCA result T_U , P_P and output, F . It is referred to as principal com-

ponent regression (PCR) and the estimated gradient by this method is:

$$\nabla F(u(k)) = P_U(T_U^T T_U)^{-1} T_U F \quad (3.60)$$

It is a combination of Eq. (3.48), least squares regression, and Eq. (3.59), PCA. The simulation results of PCR-based gradient estimation and its application to modifier adaptation are presented in Figure 9. The moving average input update is not applied, and this case uses 10 principal components and 50 data points for gradient estimation. Compared to the results from PLSR shown in Figure 3.18, PCA shows similar optimization performance of fast convergence and stability near the KKT points. However, using four principal components degraded optimization performance (the simulation results are omitted)

Another way to utilize PCA for estimating the gradient is to apply PCA to the matrix X_{UF} where the input U and output F are combined first, then the relationship between input and output is found by a missing data estimation method. The missing data estimations such as projection to the model plane (PMP), single component projection (SCP), trimmed score regression (TSR) are the methods to use the model structure or data pattern for deducing the unknown information from the known information [49]. Among these method, PMP method is utilized in this thesis because it can deduce all the unknown information at once and has computationally inexpensive compared with other missing data estimation methods [50]. When PCA is applied to the matrix X_{UF} , the loading matrix P_{UF}^T can be divided into

two parts for U and F :

$$\begin{aligned} X_{UF} &= \begin{bmatrix} U, & F \end{bmatrix} \\ \hat{X}_{UF} &= T_{UF} P_{UF}^T \\ P_{UF}^T &= \begin{bmatrix} P_U^T, & P_F^T \end{bmatrix} \end{aligned} \quad (3.61)$$

where $X_{UF} \in \mathbb{R}^{p \times n_u + n_G + 1}$, $\hat{X}_{UF} \in \mathbb{R}^{p \times n_u + n_G + 1}$, $T_{UF} \in \mathbb{R}^{p \times N_p}$, $P_{UF} \in \mathbb{R}^{n_u + n_G + 1 \times N_p}$, $P_U \in \mathbb{R}^{n_u \times N_p}$ and $P_F \in \mathbb{R}^{n_G + 1 \times N_p}$ are the input and output combined matrix, estimation by PCA, score, loading for the combined matrix and loadings for input and output, respectively. In this case, the input and output take a role of the known and missing data, respectively. PMP method estimates the missing data through the following procedure: First, score matrix by known data is estimated,

$$\hat{T}_U^T = U^T P_U (P_U P_U^T)^{-1} \quad (3.62)$$

where $\hat{T}_U \in \mathbb{R}^{p \times N_p}$ is the score matrix estimated by only the known data, U and its loading P_U ; Then the unknown output is estimated by assigning Eq. (3.61) to the model structure of PCA in Eq. (3.60),

$$\begin{aligned} \hat{F} &= P_F \hat{T}_U^T \\ &= P_F (P_U^T P_U)^{-1} P_U^T U \end{aligned} \quad (3.63)$$

where P_F is the loading matrix for the unknown data, output F . As a conclusion, the estimated gradient by PCA and PMP is:

$$\nabla F(u(k)) = P_F (P_U^T P_U)^{-1} P_U^T. \quad (3.64)$$

Unlike the PCR case in Eq. (3.60), it has a different formation as the output is considered in PCA step. The simulation results based on PCA and PMP are presented in Figure 3.19. The moving average input update is not applied and the number of principal components and data points for gradient estimation are set to be same with the PCR case. Compared to the PLSR case in Figure 3.15, it shows a similar optimization performance as PCR.

In the simulations using modifier adaptation based on PLSR and PCA, the number of principal components is fixed throughout the iterative optimization. Although using fewer principal components reduces the dimensions of the latent variable space and filters measurement noise, excessive dimension reduction may yield an inadequate description of the relationship between the input and output; as a result, it may reduce optimization performance. Thus, it is important to correctly choose the optimal number of principal components to properly explain the relationship. Generally, it is decided via N-fold cross validation [7]. For a 5-fold case, it is assumed that there are 50 data points. Then, for an arbitrary number of principal components, a latent variable space model is constructed from 40 data points, and the prediction error for the last 10 data points is calculated. These processes are performed for the last data points, and the summation of prediction errors for the number of principal components is calculated. Based on the sum of these prediction errors, the optimal value can be obtained for the current data set. The number may vary depending on the configuration of the data set, so this optimizing process should be repeated throughout the iterative optimization.

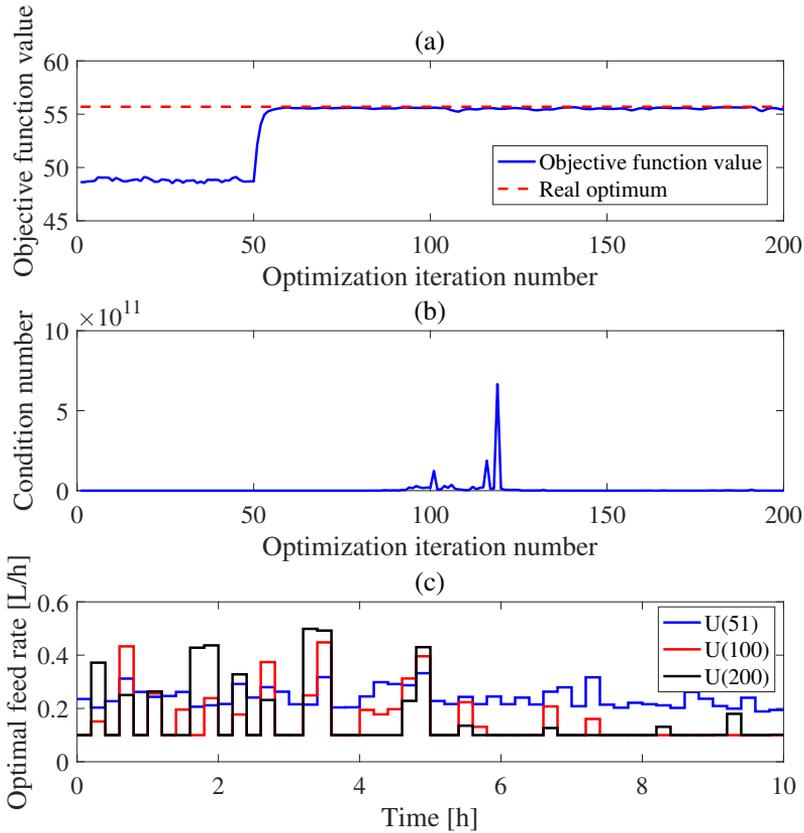


Figure 3.15: Simulation results by PLSR based gradient estimation and modifier adaptation. It is set that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation and the numbers of data points for gradient estimation and principal components are 50 and 4, respectively. The moving average input update strategy is not utilized in this case. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are $[55.5, 0.05]$ and $[55.5, 0.04]$, respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation.

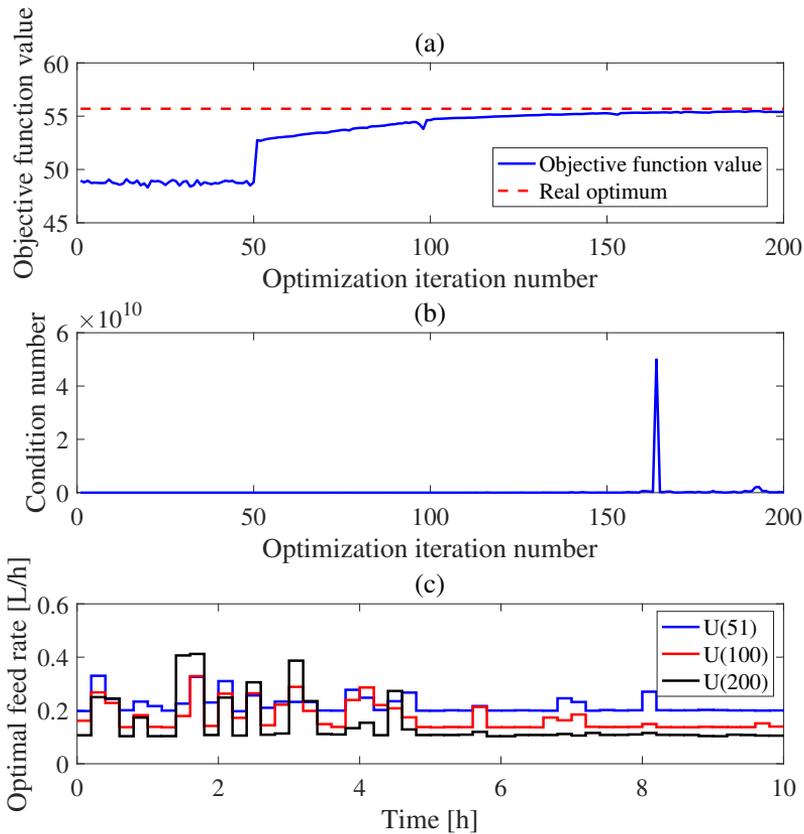


Figure 3.16: Simulation results by PLSR based gradient estimation and modifier adaptation with the moving average input update strategy. It is set that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation, the exponential filter gain for moving average input update is 0.8 and the numbers of data points for gradient estimation and principal components are 50 and 4, respectively. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are [54.8, 0.4] and [55.5, 0.6], respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation.

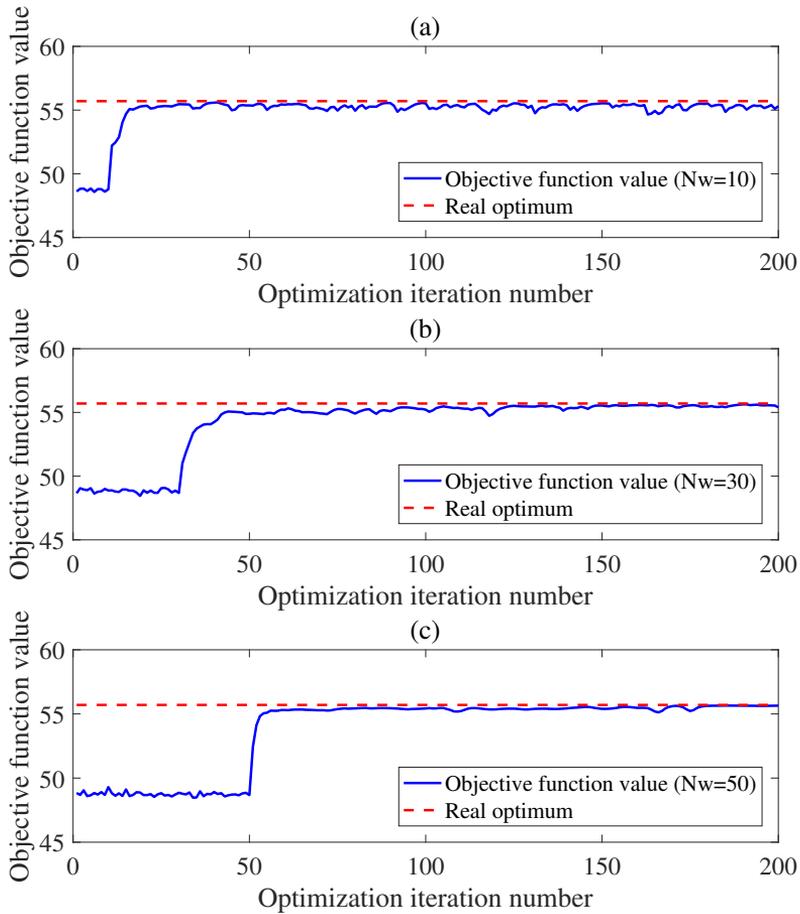


Figure 3.17: Simulation results by PLSR depending on the number of data points for gradient estimation. For emphasizing the comparison of the optimization performance near the KKT points, it is set that there exists normally distributed noise on each measurement having zero mean and 0.2 standard deviation. The numbers of data points for gradient estimation and principal components are 50 and 4, respectively. (a) Optimization performance when the number of data points for gradient estimation is 10. (b) Optimization performance when the number of data points for gradient estimation is 30. (c) Optimization performance when the number of data points for gradient estimation is 50.

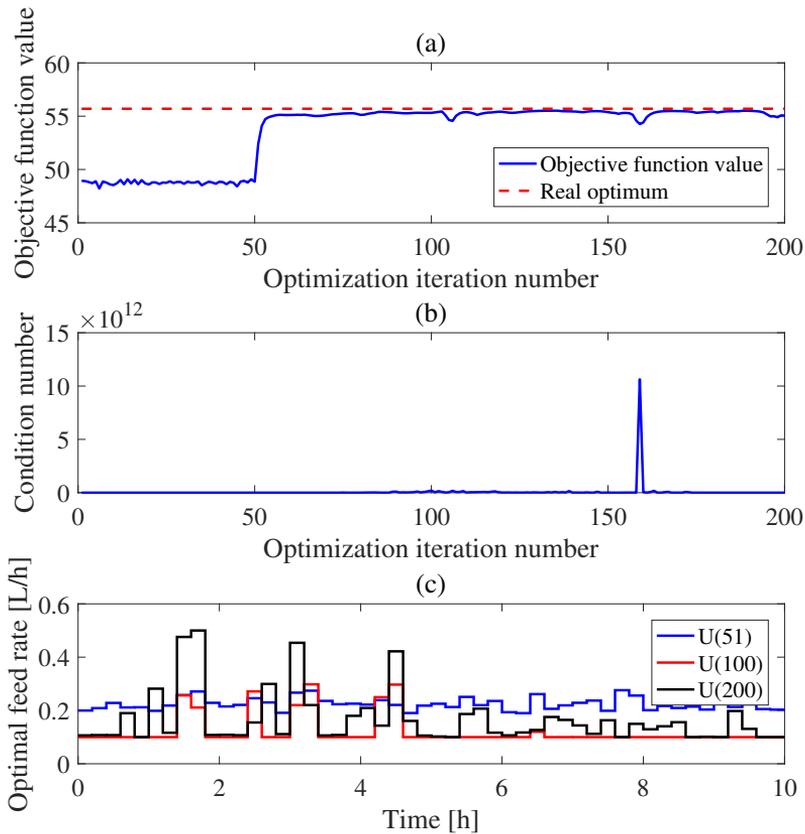


Figure 3.18: Simulation results by PCR based gradient estimation and modifier adaptation. It is set that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation and the numbers of data points for gradient estimation and principal components are 50 and 10, respectively. The moving average input update strategy is not utilized in this case. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are $[55.5, 0.08]$ and $[55.5, 0.04]$, respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation.

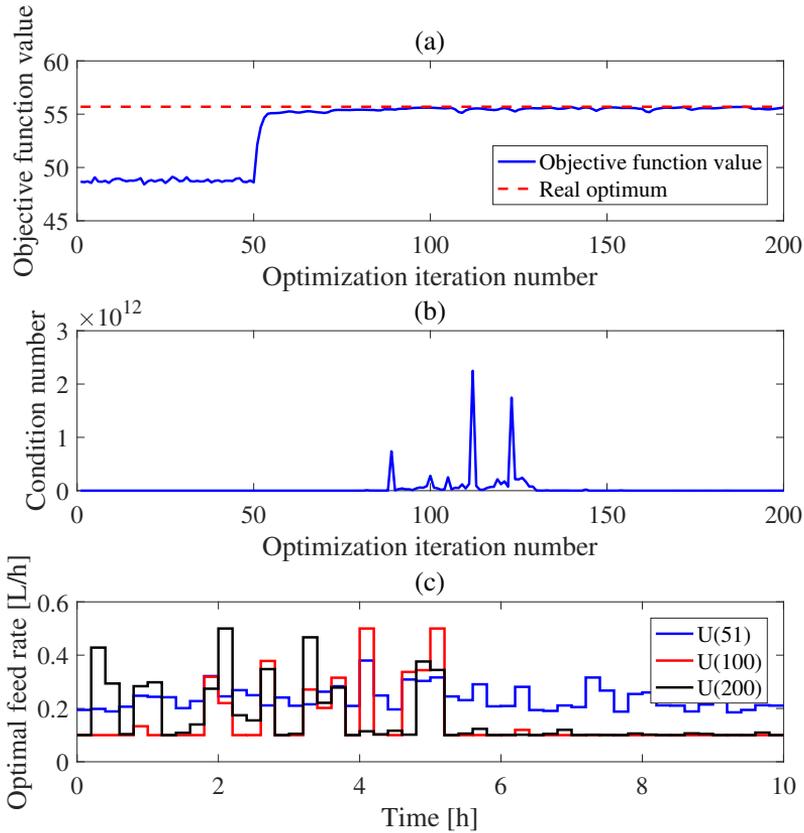


Figure 3.19: Simulation results by PCA and PMP based gradient estimation and modifier adaptation. It is set that there exists normally distributed noise on each measurement having zero mean and 0.05 standard deviation and the numbers of data points for gradient estimation and principal components are 50 and 10, respectively. The moving average input update strategy is not utilized in this case. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are $[55.5, 0.04]$ and $[55.5, 0.04]$, respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 51, 100 and 200 input updates by the modifier adaptation.

The number of data points for the regression should be set sufficiently large for better estimation performance, as shown in Figures 3.14 and 3.17. Compared to the MLR cases, PLSR exhibits fast and stable convergence to the KKT points despite using fewer data points, as shown in Figure 3.15. This property is also observed in PCA-based simulation (simulation is omitted). This occurs because correlations exist between the data, and the data points are projected to the latent variable space, which has fewer dimensions than the original space in the PLSR- and PCA-based methods. Because there are more chances for correlations to exist between the data points as the number of manipulated variables increases, latent variable space modeling can be useful for estimating the gradient in the highly multivariate case. However, utilization of more data points has the advantage of more stable convergence near the KKT points compared to using fewer data points, as shown in Figure 8. Thus, the number of data points must be adjusted according to the distance from the KKT point. When the current operating point is far from the KKT point, fewer data points should be used for fast convergence. When the current operating point is near the KKT point, more data points should be used for stable convergence.

Measurement noise affects the performance of the gradient estimation, which in turn affects the ability of the modifier adaptation scheme to converge to and satisfy the plant KKT conditions. As a result, it may make convergence to the exact KKT point impossible. In order to deal with noise having uniform distribution while optimizing the process of the modifier adaptation, the dual modifier adaptation technique was proposed by Marchetti et al. [42] based on Brdyś and Tatjewski's gradient estimation method. It claims that there are

two factors causing errors in the gradient estimation: truncation and measurement noise. By adding a constraint that considers both factors into the modifier adaptation scheme, robust input updates may be possible throughout the iterative optimization. Unlike the dual modifier adaptation case, in this study, normally distributed measurement noise is assumed in the key outputs from the simulated system. In addition, because the number of data points and principal components affects the performance of gradient estimation, the dual modifier adaptation scheme cannot be applied. However, we can expect that using more data points yields more reliable estimation by decreasing the standard deviation; as a result, the probability of convergence to the vicinity of the correct KKT point may increase. This is a key advantage that the proposed method is distinct from existing methods.

As well as the proposed moving average input update strategy in Eq. (3.49), the exponential filters in Eq. (2.2) suppress the input update. Both force data utilization by interpolation and act as smoothing filters. Although the PLSR simulation with the moving average input update whose results are shown in Figure 3.16 shows decreased optimization performance, it may be useful in cases having severe nonlinearity.

In order to reflect the optimal number of principal components and adjust the number of data points to minimize fluctuation, an algorithm is proposed:

1. $k = k + 1$
2. If $k < N_{initial}$, perturb the process and return to Step 1. Else, go to Step 3.

3. If there exists a fluctuation of the objective function value, increase the number of data points for modelling, $Nw = Nw + 1$. Else, maintain the number of data points for modelling.
4. Select the optimized number of principal components using the Nw data points and N-fold cross validation.
5. Construct a latent variable space model by using Nw data points with optimized number of principal components.
6. Calculate the modifier terms and solve the optimization problem of Eq. (3.51).
7. Implement the input by Eq. (3.51) (or Eq. (3.49) when severe non-linearity exists) to the process and get output values.
8. If the convergence condition is satisfied, Stop the algorithm. Else, return to the Step 1.

where $N_{initial}$ and Nw are the starting number of data points by perturbation and the working number of data points for gradient estimation by the latent variable space modelling, respectively. Criteria of the fluctuation in Step 3 and convergence to the KKT point are defined by each threshold value:

$$\begin{aligned}
 \text{Fluctuation: } & \frac{(\Phi(u(k-2)) - \Phi(u(k-1))) (\Phi(u(k-1)) - \Phi(u(k)))}{s_{\Phi}^2} > \alpha_{fl} \\
 \text{Convergence: } & \sum_{i=k-N_{con}+1}^k \left\| \frac{\Phi(u(i-1)) - \Phi(u(i))}{s_{\Phi}} \right\| < \alpha_{con}
 \end{aligned} \tag{3.65}$$

where Φ , u , N_{con} , s_{Φ} , α_{fl} and α_{con} are the objective function, manipulated variable, size of convergence criterion window, standard

deviation of the noise of Φ and threshold values of each criterion, respectively.

Simulation results from the proposed method are presented in Figure 3.20. For the simulation, PLSR is used for latent variable space modeling, 5-fold cross validation is applied to find the optimal number of principal components for each element, and $N_{initial}$ is set as 10. The moving average input update strategy is not utilized in this case because it only slows down the convergence. In order to show the stability around the KKT points, rather than being stopped at Step 8, the simulation is prolonged to 200 iterations. Because it has the adaptation steps for the optimal number of principal components and data points for gradient estimation, the method exhibits both fast convergence at the start of the simulation and stability near the KKT point, as shown in Figure 3.20 (a).

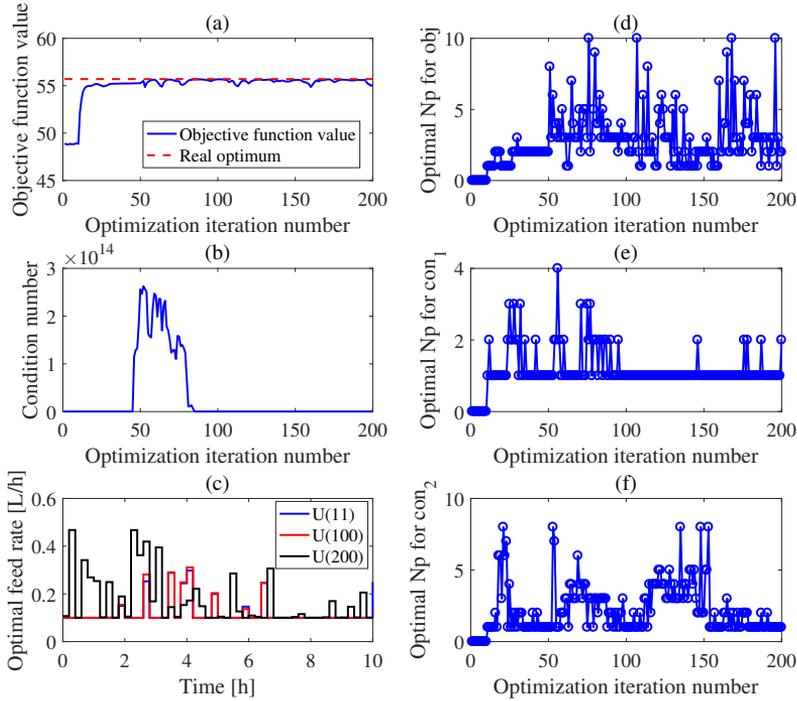


Figure 3.20: Simulation results by the proposed algorithm in Section 6.3. It is set that there exists normally distributed noise on each measurement having zero mean and 0.2 standard deviation. By the proposed method, the numbers of data points for gradient estimation and principal components are varying along the iteration of optimization. By 100 repeated simulations, the mean and variance of the objective function value at the 100th and 200th iteration are $[55.4, 0.08]$ and $[55.5, 0.07]$, respectively. (a) Optimization performance through the iterative optimization. (b) The condition numbers along the iterative optimization. (c) Calculated optimal feed rate at 11, 100 and 200 input updates by the modifier adaptation. (d) Optimal number of principal components when the output is objective function. (e) Optimal number of principal components when the output is the first inequality constraint. (f) Optimal number of principal components when the output is the second inequality constraint.

3.2.5 Concluding remarks

In this thesis, the regression-based experimental gradient estimation and its utilization within modifier adaptation are proposed for optimization of processes under the conditions of highly multivariate systems with several constraints, noisy data, and model-plant mismatch. Several regression methods, such as MLR, PLSR, and PCA, were utilized to estimate the gradient, and their optimization performance was compared with the result of Brdyś and Tatjewski's method, which is a conventional gradient estimation method. The proposed moving average input update strategy improves optimization performance by suppressing excessive input update in the case of MLR. The simulation results also show that regression-based methods, especially latent variable space modeling, outperform the other estimation methods in the case of optimizing the highly multivariate system. Although the modifier adaptation schemes based on latent variable space modeling for gradient estimation rapidly converge to the KKT point in spite of having fewer data points than the number of manipulated variables, this fewer data point strategy has the disadvantage of large and frequent fluctuations in the objective function value near the KKT point. Thus, the optimal number of principal components varies (i.e., fewer at the beginning for rapid convergence and more near the KKT point for stability) during gradient estimation with the latent variable space model. Considering this, we propose a latent variable space model-based modifier adaptation algorithm that adjusts the number of principal components and data points required for gradient estimation at each optimization step. Simulation results show that both fast convergence and stability near the KKT point are

offered by the proposed method.

3.3 Issue 3: A novel structure of modifier adaptation for robustness

3.3.1 Feasibility and structural robustness

In the process operation such as a chemical reactor with high pressure and temperature, obeying the constraints may be more important than the optimization for safety. Although there are soft constraints for operating the process in safe condition, which is defined in more conservative manner than the original constraints, they can be violated through the iterative optimization schemes like the modifier adaptation. If we can guarantee the feasibility of real process through the proposed method, the available operating region can be widened by easing the soft constraints and optimization performance can be improved. In order to address the feasibility issue and structural robustness, we propose a new modifier adaptation approach which guarantees to provide a sequence of feasible inputs during iterations and conservativeness to the input update. The proposed modifiers are not zero or first-order ones like the conventional modifier adaptation, but are the coefficients of high-order terms approximating the plant equations. By allowing the order of approximate equations to be higher than the first-order, the iterated inputs are guaranteed to respect the constraints and to reach a KKT point of plant. Moreover, the convexity of approximate high-order equations automatically ensures the model adequacy, which is the necessary condition of convergence for the modifier adaptation scheme.

Assumption 3.1. *The objective and constraint equations of plant are smooth or have derivatives of all orders everywhere in their feasible regions.*

Assumption 3.2. *The modifier adaptation schemes have at least one fixed point as solution to optimization problem.*

Assumption 3.3. *Gradient estimates of the objective and constraints are accurate.*

Example 1. The optimization problem for plant equations are:

$$\begin{aligned}
 \min_{\mathbf{u}} \quad & \Phi_p := (u_1 - 1)^2 + (u_2 - 1)^2 + (u_3 + 1)^2 \\
 & + (u_4 + 2)^2 + (u_5 - 2)^2 + (u_6 - 1)^2 \\
 \text{s.t.} \quad & G_{p,1} := 2u_1 + 3u_2 - u_3 - 2u_4 + u_5^2 - 2u_6 - 5 \leq 0 \\
 & G_{p,2} := u_1^2 + 2u_2 - u_3^2 - 2u_4 + 3u_5 - 2u_6 - 6 \leq 0.
 \end{aligned} \tag{3.66}$$

, which has the optimal solution of

$$\mathbf{u}_p^* = [0.55, 0.33, -0.78, -1.55, 1.38, 1.45]^T$$

. The model equations and related optimization problem are given as:

$$\begin{aligned}
 \min_{\mathbf{u}} \quad & \Phi_m := (u_1 - 1.5)^2 + (u_2 - 1.7)^2 + (u_3 + 1.9)^2 \\
 & + (u_4 + 2.5)^2 + (u_5 - 2.2)^2 + (u_6 - 1.8)^2 \\
 \text{s.t.} \quad & G_{m,1} := 2.9u_1 + 3.9u_2 - 1.9u_3 - 2.9u_4 + 2.9u_5^2 \\
 & - 2.9u_6 - 1 \leq 0 \\
 & G_{m,2} := 1.5u_1^2 + 2.5u_2 - 0.8u_3^2 - u_4 + 3u_5 \\
 & - 2.1u_6 - 2 \leq 0.
 \end{aligned} \tag{3.67}$$

Its solution,

$$\mathbf{u}_m^* = [0.38, 0.20, -1.17, -1.38, 0.68, 2.92]^T$$

is different from the above real optimal solution of plant. When the conventional modifier adaptation scheme Eq. (2.1) is applied to this problem, it can find the real plant optimum as shown in Figure 3.21 upon convergence. However, in the process of iterative input updates, it brings a few violations of constraints as described in Figure 3.22.

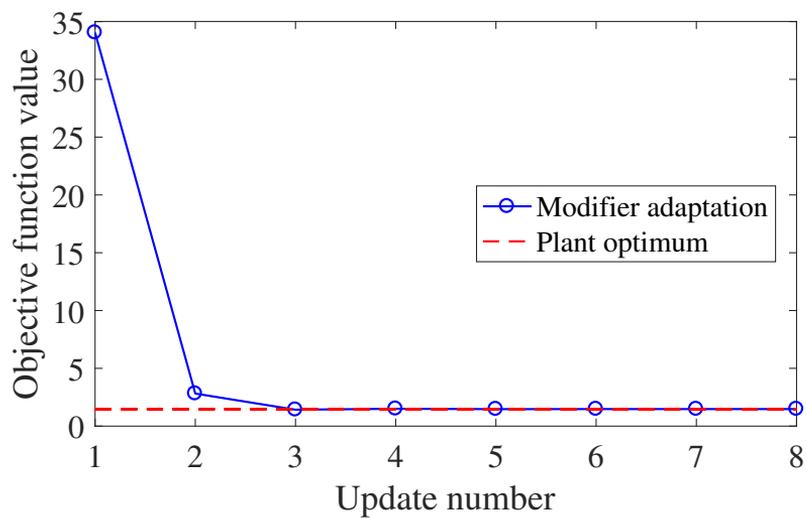


Figure 3.21: Update of objective function value by the scheme of the conventional modifier adaptation in Eq. (2.1).

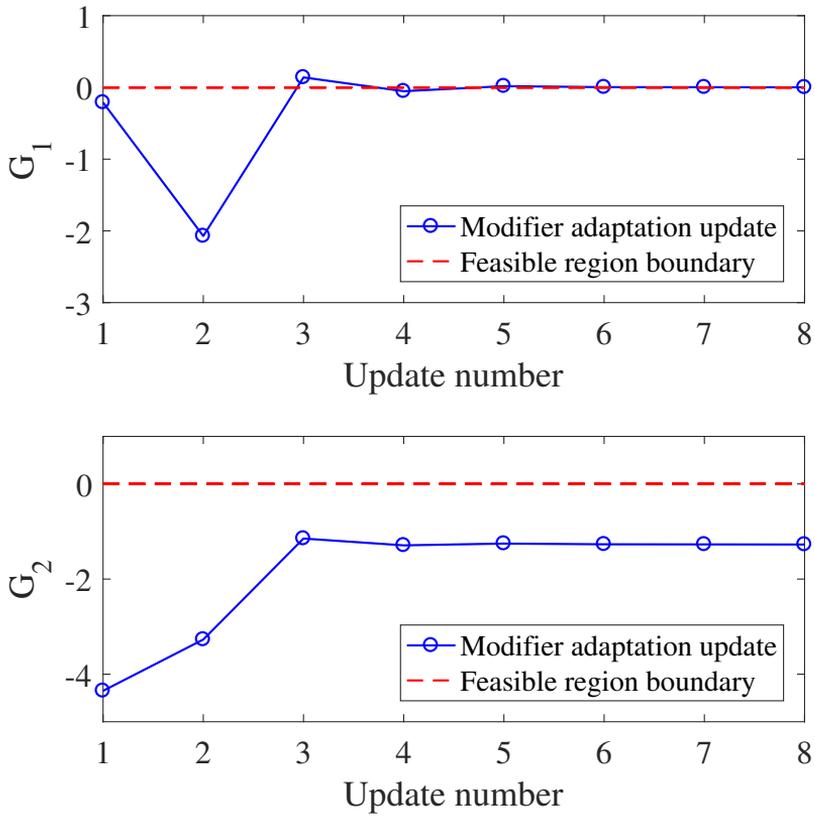


Figure 3.22: Update of constraint functions by the scheme of the conventional modifier adaptation in Equ. (2.1).

3.3.2 Proposition of new structural modifier adaptation

In order to overcome the infeasible path update of the conventional modifier adaptation, a new formulation is proposed as follows:

(Proposition)

$$\begin{aligned} u^*(k+1) &:= \arg \min_u \Phi_m(u) := u_m^{\Phi T} R_{\Phi} u_m^{\Phi} + \xi^{\Phi} \\ \text{s.t. } G_m(u) &:= u_m^{G T} R_G u_m^G + \xi^G \leq 0 \end{aligned} \quad (3.68)$$

where the related variables, coefficients and modifiers are defined as follows:

$$u_{m,i}^F := (u_i - \gamma_i^F)^w \quad (3.69a)$$

$$R_F := \begin{bmatrix} \varepsilon_1^F & 0 & \cdots & 0 \\ 0 & \varepsilon_2^F & \cdots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \varepsilon_n^F \end{bmatrix} \quad (3.69b)$$

$$\gamma_i^F := \begin{cases} \text{if } \nabla F_{p,i}(u^{*,r}(k)) \geq 0, \\ u_i^{*,r}(k) - \left(\frac{\nabla F_{p,i}(u^{*,r}(k))}{2w\varepsilon_i^F} \right)^{\frac{1}{2w-1}} \\ \text{else} \\ u_i^{*,r}(k) + \left(\frac{|\nabla F_{p,i}(u^{*,r}(k))|}{2w\varepsilon_i^F} \right)^{\frac{1}{2w-1}} \end{cases} \quad (3.69c)$$

$$\xi^F := F_p(u^{*,r}(k)) - \sum_{i=1}^n \varepsilon_i^F (u_i^{*,r}(k) - \gamma_i^F)^{2w} \quad (3.69d)$$

where $F(u)$ is a functional set consisting of objective, Φ and constraints, G , that is, $F(u) := \{u | \Phi(u) = 0 \text{ or } G(u) = 0\}$. u_i and

$u_i^{*,r}(k)$ are the i^{th} elements of vectors $u \in \mathbb{R}^{n \times 1}$ and $u^{*,r}(k) \in \mathbb{R}^{n \times 1}$, respectively. $u_m^F \in \mathbb{R}^{n \times 1}$ is a vector consisting of $u_{m,i}^F$. $\gamma_i^F \in \mathbb{R}$ and $\xi^F \in \mathbb{R}$ are the newly defined modifiers of the proposed method. $w \in \mathbb{N}$ and $\varepsilon_i^F \in \mathbb{R}^+$ are the coefficients, which can be arbitrarily adjusted prior to the iterative update of input. $n \in \mathbb{N}$ and $k \in \mathbb{N}$ are the numbers of input variables and updates, respectively.

In the iterative optimization and input update, $u^*(k+1)$ from the proposition is not implemented directly, but used to calculate $u^{*,r}(k+1)$ for implementing on the real system, in order to avoid a sudden change in the input.

$$u^{*,r}(k+1) := (1 - \alpha)u^{*,r}(k) + \alpha u^*(k+1) \quad (3.70)$$

Note that it is possible to set α as a sequence of real numbers in $[0, 1]$ satisfying $\sum_{i \in \mathbb{N}} \alpha(i)(1 - \alpha(i)) = +\infty$ in general [10]. However, in this work the α is set as a constant real number in $(0, 1)$ without loss of generality.

Theorem 3.4. *At the k^{th} optimal input $u^{*,r}(k)$, the values and gradients of the modified model equations $F_m(u)$ for $u^*(k+1)$ (Proposition) are the same with those of the plant equations $F_p(u)$ in Eq. (2.1).*

$$F_m(u^{*,r}(k)) = F_p(u^{*,r}(k)) \quad (3.71a)$$

$$\nabla F_m(u^{*,r}(k)) = \nabla F_p(u^{*,r}(k)) \quad (3.71b)$$

If the index w or coefficient ε_i^F for the modified model is set suf-

sufficiently large, the following holds:

$$F_m(u) \geq F_p(u), \forall u = \{u | G_p(u) \leq 0\} \quad (3.72)$$

Proof Substitution of Eq (3.69d) into $F_m(u^*(k))$ gives

$$\begin{aligned} & F_m(u^{*,r}(k)) \\ &= \sum_{i=1}^n \varepsilon_i^F (u_j^{*,r}(k) - \gamma_i^F)^{2w} + \xi^F \\ &= \sum_{i=1}^n \varepsilon_i^F (u_i^{*,r}(k) - \gamma_i^F)^{2w} + F_p(u^{*,r}(k)) \\ &\quad - \sum_{i=1}^n \varepsilon_i^F (u_j^{*,r}(k) - \gamma_i^F)^{2w} \\ &= F_p(u^{*,r}(k)) \end{aligned} \quad (3.73)$$

where γ_j^F is from Eq. (3.69c). Substitute Eq. (3.69c) into $\nabla F_{p,i}(u^{*,r}(k))$, $\forall i \in \{1, \dots, n\}$. Then for the index i satisfying $\nabla F_{p,i}(u^{*,r}(k)) \geq 0$, the following holds:

$$\begin{aligned} & \nabla F_{m,i}(u^{*,r}(k)) \\ &= 2w\varepsilon_i^F (u_j^{*,r}(k) - \gamma_i^F)^{2w-1} \\ &= 2w\varepsilon_i^F \left(u_j^{*,r}(k) - u_j^{*,r}(k) + \left(\frac{\nabla F_{p,i}(u^{*,r}(k))}{2w\varepsilon_i^F} \right)^{\frac{1}{2w-1}} \right)^{2w-1} \\ &= \nabla F_{p,i}(u^{*,r}(k)) \end{aligned} \quad (3.74)$$

The case, in which $\nabla F_{p,i}(u^{*,r}(k)) < 0$, is omitted because it is similar with the above case. As a conclusion, the values and gradients of modified model equations $F_m(u)$ for $u^*(k+1)$ are equal to those of

the plant equations $F_p(u)$ at the k^{th} optimal input $u^{*,r}(k)$.

On the other hand, approximation of the plant equations $F_p(u)$ by its Taylor expansion $F_{p,a}(u)$ at $u^{*,r}(k)$ is given by:

$$\begin{aligned}
F_p(u) &\cong F_{p,a}(u) \\
&= F_p(u^{*,r}(k)) + \nabla F_p(u^{*,r}(k))^T (u - u^{*,r}(k)) \\
&\quad + \frac{1}{2!} (u - u^{*,r}(k))^T \nabla^2 F_p(u^{*,r}(k)) (u - u^{*,r}(k)) \\
&\quad + H.O.T.(u)
\end{aligned} \tag{3.75}$$

For an arbitrary input deviation Δu , if it is subtracted from the modified model F_m at $(u^{*,r}(k) + \Delta u)$, the following equation holds:

$$\begin{aligned}
&F_m(u^{*,r}(k) + \Delta u) - F_{p,a}(u^{*,r}(k) + \Delta u) \\
&= \sum_{i=1}^n \varepsilon_i^F \left(\Delta u + \left(\frac{\nabla F_{p,i}(u^{*,r}(k))}{2w\varepsilon_i^F} \right)^{\frac{1}{2w-1}} \right)^{2w} \\
&\quad - \sum_{i=1}^n \varepsilon_i^F \left(\left(\frac{\nabla F_{p,i}(u^{*,r}(k))}{2w\varepsilon_i^F} \right)^{\frac{1}{2w-1}} \right)^{2w} \\
&\quad - \nabla F_p(u^{*,r}(k))^T \Delta u - \frac{1}{2!} \Delta u^T \nabla^2 F_p(u^{*,r}(k)) \Delta u \\
&\quad + H.O.T.(\Delta u)
\end{aligned} \tag{3.76}$$

Since the case of $w > 1$ is similar, we only consider the case of

$w = 1$ for simplicity.

$$\begin{aligned}
& F_{m|k=1}(u^{*,r}(k) + \Delta u) - F_{p,a}(u^{*,r}(k) + \Delta u) \\
&= \sum_{i=1}^n \varepsilon_i^F \left(\Delta u + \frac{\nabla F_{p,i}(u^{*,r}(k))}{2\varepsilon_i^F} \right)^2 \\
&\quad - \sum_{i=1}^n \varepsilon_i^F \left(\frac{\nabla F_{p,i}(u^{*,r}(k))}{2\varepsilon_i^F} \right)^2 \\
&\quad - \nabla F_p(u^{*,r}(k))^T \Delta u - \frac{1}{2!} \Delta u^T \nabla^2 F_p(u^{*,r}(k)) \Delta u \\
&\quad - H.O.T.(\Delta u) \tag{3.77} \\
&= \Delta u^T \left(\begin{bmatrix} \varepsilon_1^F & 0 & \cdots & 0 \\ 0 & \varepsilon_2^F & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \varepsilon_n^F \end{bmatrix} - \frac{1}{2!} \nabla^2 F_p(u^{*,r}(k)) \right) \Delta u \\
&\quad - H.O.T.(\Delta u) \\
&= \Delta u^T M \Delta u - H.O.T.(\Delta u)
\end{aligned}$$

where M is a matrix function of ε_i^F and $u^{*,r}(k)$. For all $u^{*,r}(k) \in \{u | G_p(u) \leq 0\}$ and $\Delta u \in \Delta U : \{\Delta u | G_p(u^{*,r}(k) + \Delta u) \leq 0, \Delta u \neq 0\}$, it is possible to make Eq. (3.77) positive by adjusting the modifier coefficient ε_i^F sufficiently large while satisfying Eq. (3.71a-b). When $H.O.T.(\Delta u)$ is small enough to ignore, it is possible to make Eq. (3.77) positive by adjusting M to be positive definite. ■

Remark 1. For the feasible update of input pathway, the index w or coefficient ε_i^F should be set large enough to satisfy Eq. (3.77). However, as larger values of w and ε_i^F are set, the form of modified model becomes more acute or has a large curvature, and as a result, the feasible region of the modified model becomes more nar-

row. Thus, the convergence rate of the proposed modifier adaptation decreases when w and ε_i^F increase.

Remark 2. Unlike the previous methods for the feasible update of input pathway [11, 34], the proposed method does not suffer from multiple constraints or post-processing problem. It preferentially adjusts the coefficients w and ε_i^F of the modified model to be sufficiently large and yields a feasible update pathway during iterative optimization.

Corollary 3.2. *In the proposed modifier adaptation Eq. (3.68, 69a-d), let the modifiers be determined as Eq. (3.69a-d). Let the feasible regions of the modified model and plant be defined as:*

$$\Psi_m := \{u | G_m(u) \leq 0\} \quad (3.78a)$$

$$\Psi_p := \{u | G_p(u) \leq 0\} \quad (3.78b)$$

Then if the index w or coefficient ε_i^F are set sufficiently large to satisfy Eq. (3.71a), the following relation holds:

$$\Psi_m \subset \Psi_p \quad (3.79)$$

Proof By **Theorem 3.4**, the following relation holds: if $G_m(u) \leq 0$, then $G_p(u) \leq 0$. Thus the relation Eq. (3.79) is true. ■

Remark 3. **Corollary 3.2** implies that the feasible region of plant includes that of the modified model by setting the coefficients sufficiently large. This guarantees that the solution of Proposition, $u^*(k+1)$, is feasible for the plant Eq. (1.1). The input for implementation $u^{*,r}(k+1)$, which is a linear combination of $u^{*,r}(k)$ and $u^*(k+1)$ as shown in Eq. (3.70), is also feasible the plant Eq. (1.1).

Thus, solving the proposition problem and iteratively updating input with Eq. (3.70) ensures that the updated input pathway $u^{*,r}(k)_{k \in \mathbb{N}}$ is feasible for the plant constraints while the iterative updates.

Corollary 3.3. *The proposed modifier adaptation scheme is a strictly convex optimization problem, which is also adequate.*

Proof Since the coefficients w and ε_i^F are set to be positive for the approximation, both of the objective and constraint equations in Eq. (3.68) are strictly convex. If the modifiers can be found such that a fixed point of the modifier adaptation scheme is equal to the plant optimum, the model adequacy holds [5]. As proven by François et al., if the modified model is strictly convex, it is adequate [1]. ■

Remark 4. If the model is inadequate, no values of modifier can be found such that plant optimum corresponds to an optimum of the modified model, by the definition of model adequacy [5]. Thus, the model adequacy is a necessary condition of convergence to a KKT point of plant.

Theorem 3.5. *If the proposed modifier adaptation scheme has at least one fixed point as the solution to Eq. (3.68), the updated input sequence converges to a fixed point during the iterative optimization.*

Proof As proven in Faulwasser et al. [10], the sufficient conditions of convergence by modifier adaptation are as follows:

- 1 The objective and constraints are twice continuously differentiable in both of the plant and modified model.

- 2 The modifier adaptation scheme has at least one fixed point as an optimal solution.
- 3 The modifier adaptation scheme is globally feasible for the plant constraints.
- 4 The modifier adaptation scheme has a unique solution.
- 5 The modifier adaptation scheme is nonexpansive.

The first and second conditions are satisfied by **Assumption 3.1** and **Assumption 3.2**. The third condition is satisfied by **Corollary 3.2**, which ensures the feasible input pathway during the iterative optimization. The fourth condition is satisfied by **Corollary 3.3** which ensures that the modified optimization can be made strictly convex by adjusting the modifier coefficients w and ε_i^F to be positive. Thus, the solution is unique. Finally, the fifth condition is satisfied by the fourth sufficient condition because of the definition of the following nonexpansive map:

$$\forall x, y \in C \quad \|H(x) - H(y)\| \leq \|x - y\| \quad (3.80)$$

where $H : C \rightarrow C$ is a map and $C \subset \mathbb{R}^n$ is a nonempty convex set. When the proposed modifier adaptation scheme is seen as a mapping such that $u^* : u \mapsto u^*$, it has a solution (the second condition) and it is unique (the fourth condition). Thus the proposed modifier adaptation scheme is a nonexpansive mapping. As a result, the sequence $(u^{*,r}(k) - u(k))_{k \in \mathbb{N}}$ converges to 0 by the Krasnoselski-Mann theorem for the averaged operator [10]. ■

Theorem 3.6. *Upon convergence, the converged solution of the proposed modifier adaptation scheme satisfies the KKT conditions of plant equations.*

Proof In the whole process of iterative optimization, the values and gradients of modified model $F_m(u)$ are equal to those of the plant $F_p(u)$ of Eq. (1.1), as expressed in Eq. (3.71a-b) of **Theorem 3.4**. When the converged input is defined as $u_{m,\infty}^{*,r}$, the following equations hold:

$$F_m(u_{\infty}^{*,r}) = F_p(u_{\infty}^{*,r}) \quad (3.81a)$$

$$\nabla F_m(u_{\infty}^{*,r}) = \nabla F_p(u_{\infty}^{*,r}) \quad (3.81b)$$

Thus KKT conditions in Eq. (1.2a-d) are satisfied at $u_{\infty}^{*,r}$ by replacing $F_m(u_{\infty}^{*,r})$ and $\nabla F_m(u_{\infty}^{*,r})$ with $F_p(u_{\infty}^{*,r})$ and $\nabla F_p(u_{\infty}^{*,r})$, respectively.

■

Remark 5. By **Theorem 3.5**, the converged solution by the proposition is the KKT point of the plant equation as well as the modified equations upon convergence. However, it is not a sufficient condition for (local) optimality of plant, but just a necessary condition. In order to reach a local optimum, an additional assumption about second-order derivatives should be satisfied as described in the following **Corollary 3.4**. For this, related concepts of a Lagrangian function $L_m : \mathbb{R}^n \times \mathbb{R}^{\hat{j}} \rightarrow \mathbb{R}$ are defined as

$$L_m(u, \mu_j) := \Phi_m(u) + \sum_{j \in J} \mu_j G_{m,j}(u) \quad (3.82)$$

where $J \subset \mathbb{N}$, μ and \hat{j} denote the set of active constraints, the La-

grange multipliers and the cardinality of J at u_∞^* , respectively.

Corollary 3.4. *Upon convergence, if the converged point satisfies the strict second-order sufficient condition of optimality as follow:*

$$\forall q \in C_m(u_\infty^*) : q^T \frac{\partial^2 L_m}{\partial u^2} \Big|_{u_\infty^*} q > 0 \quad (3.83)$$

where $C_m \subset \mathbb{R}^n$ and u_∞^* are corresponding cone and a local minimal point of plant equation, then $u_\infty^{*,r}$ from the iterative optimization of Eq. (3.68) is a local minimizer of Eq. (1.1).

Proof By **Theorem 3.6**, the converged solution $u_\infty^{*,r}$ is a KKT point of the plant Eq. (1.1). If Eq. (3.83) is satisfied additionally, it is a local minimizer of Problem 1 by the second order sufficient condition of optimality [1, 10]. ■

Example 2. The same optimization problem of **Example 1** is solved by the proposed modifier adaptation scheme in Eq. (3.68) and Eq. (3.69a-d). The simulation results are described in Figures 3.23 and 3.24. For simulation, both the coefficients w and ε were defined as 1. By the proposed method with predefined sufficiently large coefficient values, the real plant optimum could be found without violating the multiple constraints as described in Figures 3.21 and 3.22. Because of the large coefficient values, the rate of convergence for the proposed method is slower than that of the conventional method. As the coefficient values decrease, the rate of convergence becomes faster.

When the model equations for constraint in the conventional modifier adaptation has an order less than 1, the feasible region is affected linearly by the first-order modifier term. However, in the proposed

novel structure of modifier adaptation, the effects of error in functional measurement and gradient estimation can be suppressed structurally.

Theorem 3.7. *The error in functional measurement and gradient estimation is order of $1/2w$ by the proposed structure of modifier adaptation in Eq. (3.68) and Eq. (3.69a-d) when the values of w and ε are set sufficiently large.*

Proof Without loss of generality, 1-dimensional problem of the proposed modifier adaptation is considered. The solution of constraint function of 1-dimensional case when constraint value is G and the gradient value is $\frac{\partial G}{\partial u}$ at u_k :

$$\begin{aligned} u^+ &= u_k - \left(\frac{\partial G / \partial u}{2w\varepsilon^F} \right)^{1/(2w-1)} + \left(\left(\frac{\partial G / \partial u}{2w\varepsilon^F} \right)^{2w/(2w-1)} - \frac{G}{\varepsilon^F} \right)^{1/2w} \\ u^- &= u_k - \left(\frac{\partial G / \partial u}{2w\varepsilon^F} \right)^{1/(2w-1)} - \left(\left(\frac{\partial G / \partial u}{2w\varepsilon^F} \right)^{2w/(2w-1)} - \frac{G}{\varepsilon^F} \right)^{1/2w} \end{aligned} \quad (3.84)$$

When the errors of G and $\frac{\partial G}{\partial u}$ are small and have the values of G_e and $\frac{\partial G}{\partial u}_e$ and the values of w and ε are set sufficiently large, the solution of the constraint (u^+ and u^-) can be considered as a function of order $1/2w$. It is because $2w/(2w-1) \simeq 1$ and $x^{1/(2w-1)} < x^{1/2w}$ when x

is sufficiently small:

$$\begin{aligned}
u^+ &= u_k - \left(\frac{\partial G / \partial u}{2w\varepsilon^F} \right)^{1/(2w-1)} + \left(\left(\frac{\partial G / \partial u}{2w\varepsilon^F} \right)^{2w/(2w-1)} - \frac{G}{\varepsilon^F} \right)^{1/2w} \\
&\simeq u_k - \left(\frac{\partial G / \partial u}{2w\varepsilon^F} \right)^{1/(2w-1)} + \left(\frac{\partial G / \partial u}{2w\varepsilon^F} - \frac{G}{\varepsilon^F} \right)^{1/2w} \\
u^+ - u_e^+ &= O(G_e^{1/2w}, \left(\frac{\partial G}{\partial u} \right)_e^{1/2w})
\end{aligned} \tag{3.85}$$

where u_e^+ is the function of measurement and gradient error. The function of u^- has the same order of $G_e^{1/2w}, \left(\frac{\partial G}{\partial u} \right)_e^{1/2w}$, thus is omitted here. ■

As a conclusion, with the new structure of modifier adaptation as Eq. (3.68-69), structural robustness to the measurement and gradient estimation noise can be achieved.

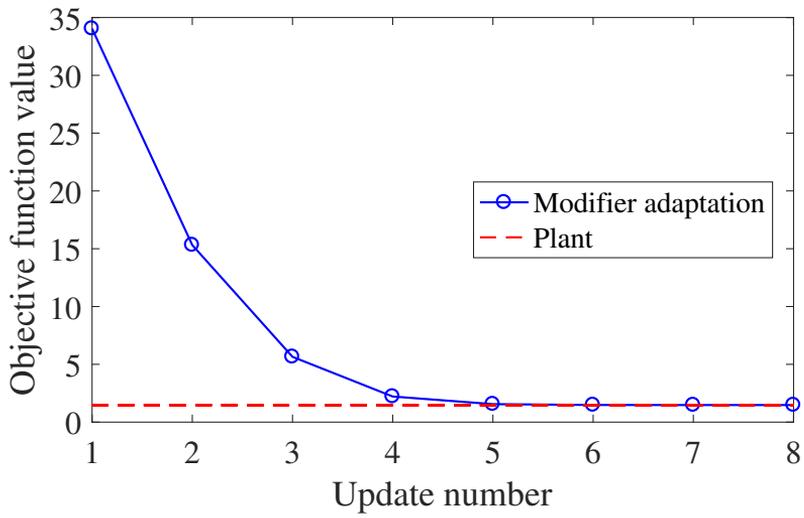


Figure 3.23: Update of objective function value by the scheme of the proposed modifier adaptation in Eq. (3.68) and Eq. (3.69a-d).

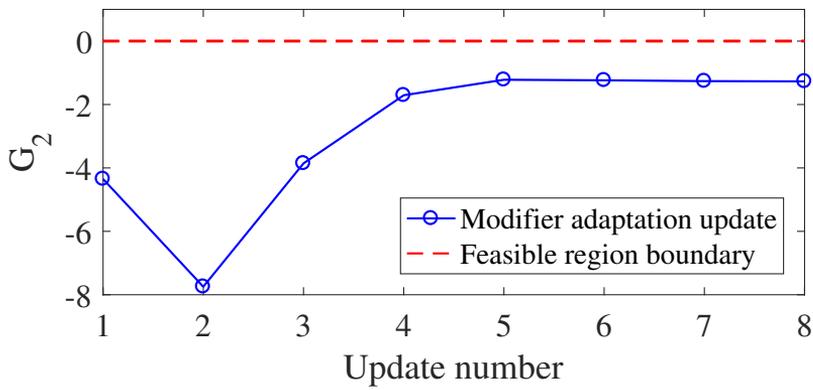
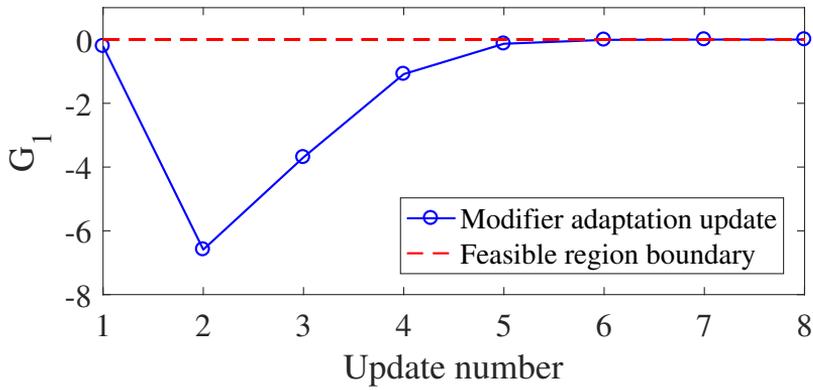


Figure 3.24: Update of constraint functions by the scheme of the proposed modifier adaptation in Eq. (3.68) and (3.69a-d).

3.3.3 Illustrative example

In this section, an isothermal continuous stirred-tank reactor at steady state is optimized, in which the reactions $A + B \xrightarrow{k_1} P$ and $B + P \xrightarrow{k_2} R$ take place. A and B are reactants, P is the desired product and R is the residual, respectively. Mass balance equations are:

$$\begin{aligned}\frac{dC_A}{dt} &= -k_1 C_A C_B + \frac{u_A C_{A,in}}{V} - \frac{(u_A + u_B) C_A}{V} \\ \frac{dC_B}{dt} &= -k_1 C_A C_B - k_2 C_B C_P + \frac{u_B C_{B,in}}{V} \\ &\quad - \frac{(u_A + u_B) C_B}{V} \\ \frac{dC_P}{dt} &= k_1 C_A C_B - k_2 C_B C_P - \frac{(u_A + u_B) C_P}{V} \\ \frac{dC_R}{dt} &= k_2 C_B C_P - \frac{(u_A + u_B) C_R}{V}\end{aligned}\tag{3.86}$$

where C_A , C_B , C_P and C_R are the concentrations of each component and k_1 , k_2 , u_A , u_B , V , $C_{A,in}$ and $C_{B,in}$ are the rate constants of each reaction, the input flow rates A and B , the operating volume and the concentration of each input flow rate, respectively. The optimization

problem, where the steady state is assumed, is formulated as follows:

$$\begin{aligned}
\max_{\bar{u}_A, \bar{u}_B} \Phi &:= w_P \bar{C}_P (\bar{u}_A + \bar{u}_B) - w_R \bar{C}_R (\bar{u}_A + \bar{u}_B) \\
\text{s.t.} \quad &-k_1 \bar{C}_A \bar{C}_B + \frac{\bar{u}_A C_{A,in}}{V} - \frac{(\bar{u}_A + \bar{u}_B) \bar{C}_A}{V} = 0 \\
&-k_1 \bar{C}_A \bar{C}_B - k_2 \bar{C}_B \bar{C}_P + \frac{\bar{u}_B C_{B,in}}{V} \\
&- \frac{(\bar{u}_A + \bar{u}_B) \bar{C}_B}{V} = 0 \\
&k_1 \bar{C}_A \bar{C}_B - k_2 \bar{C}_B \bar{C}_P - \frac{(\bar{u}_A + \bar{u}_B) \bar{C}_P}{V} = 0 \\
&k_2 \bar{C}_B \bar{C}_P - \frac{(\bar{u}_A + \bar{u}_B) \bar{C}_R}{V} = 0 \tag{3.87} \\
P_R &:= \frac{\bar{C}_R}{\bar{C}_A + \bar{C}_B + \bar{C}_P + \bar{C}_R} \\
Q &:= k_1 \bar{C}_A \bar{C}_B V (-\Delta H_1) - k_2 \bar{C}_A \bar{C}_B V (-\Delta H_2) \\
P_R - P_{R,max} &\leq 0 \\
Q - Q_{max} &\leq 0 \\
\bar{u}_{A,min} &\leq \bar{u}_A \leq \bar{u}_{A,max} \\
\bar{u}_{B,min} &\leq \bar{u}_B \leq \bar{u}_{B,max}
\end{aligned}$$

where Φ is the objective function which maximizes the production rate of P while minimizing that of R and w_P , and w_R are the weighting factors. \bar{C}_A , \bar{C}_B , \bar{C}_P and \bar{C}_R are the concentrations of each components at steady state, \bar{u}_A and \bar{u}_B are the input flow rates of A and B to be optimized, P_R and $P_{R,max}$ are the proportion of R and its upper bound, Q and Q_{max} are the generated heat and its upper bound, and $\bar{u}_{A,min}$, $\bar{u}_{A,max}$, $\bar{u}_{B,min}$ and $\bar{u}_{B,max}$ are the lower and upper bounds of the input flow rates, respectively. The nominal values of parameters for simulation are given in Table 3.4.

The simulation results by the proposed modifier adaptation scheme are presented in Figure 3.25 and 3.26. It could find the real plant optimum despite the model-plant mismatch without violating the multiple constraints, which generally fails in the conventional methods. If one sets the coefficient values w and ε lower than the values denoted in Table 3.4, faster convergence to the real plant optimum can be achieved. However, when it is outside the threshold, that is, the curvature of plant equation is larger than that of the modifier adapted equations of Eq. (3.68) and Eq. (1.1), it may bring infeasible input updates during the optimization. Thus, for guaranteeing not to violate the constraints, the coefficients should be set sufficiently large.

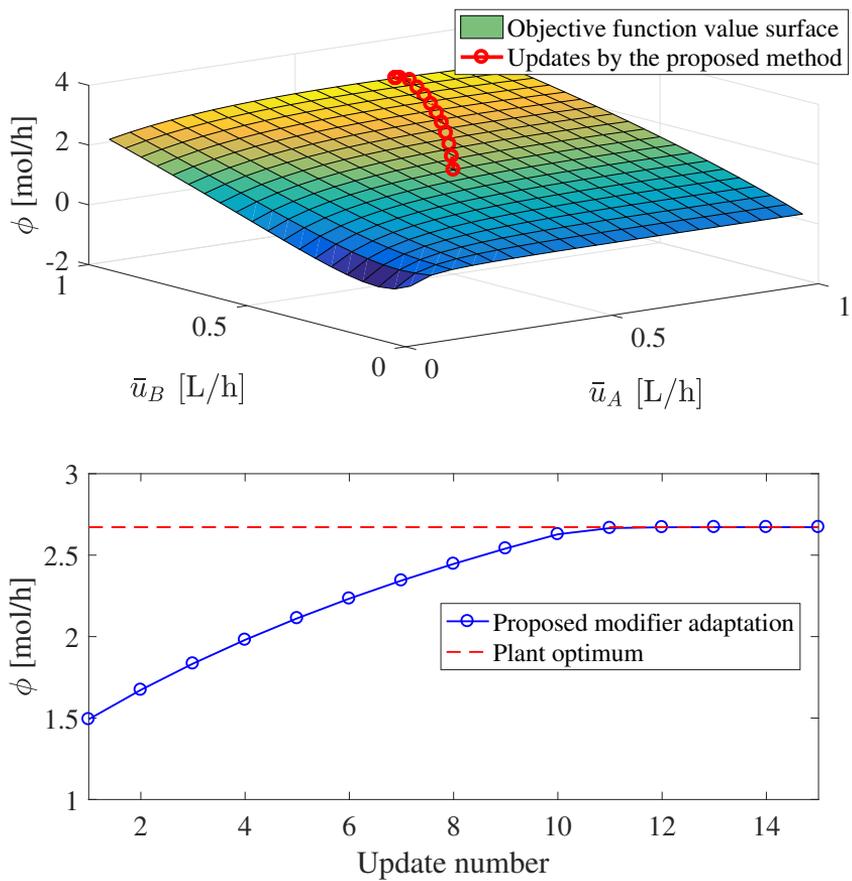


Figure 3.25: Update of objective function of the continuous stirred-tank reactor example by the scheme of the proposed modifier adaptation in Eq. (3.68) and Eq. (3.69a-d).

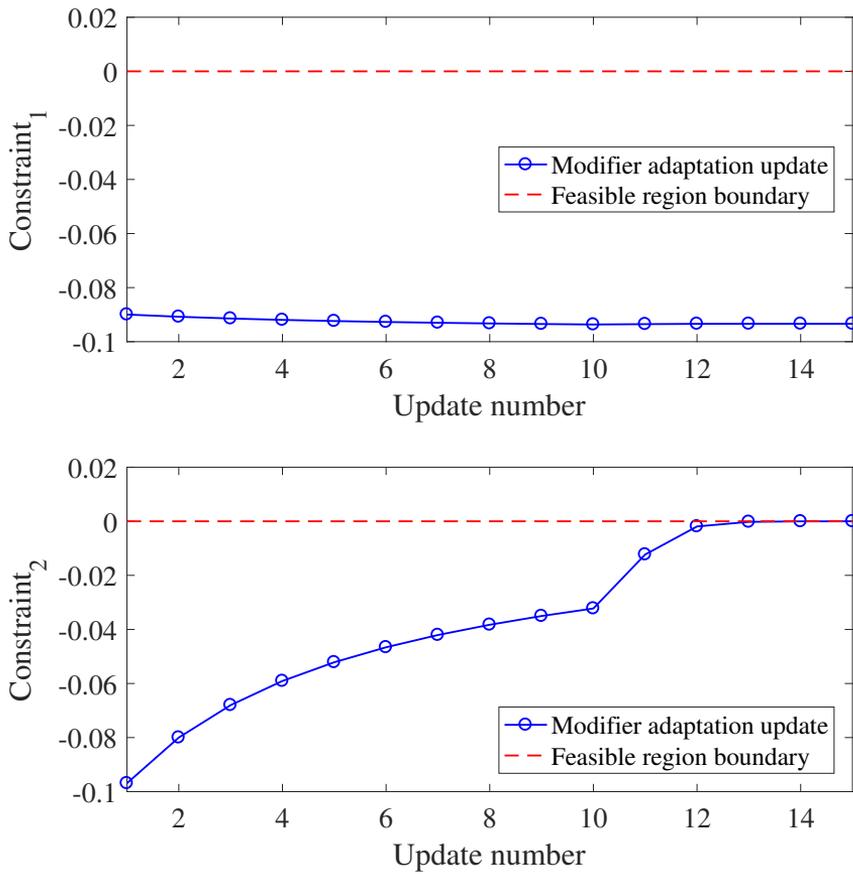


Figure 3.26: Update of constraint functions of the continuous stirred-tank reactor example by the scheme of the proposed modifier adaptation in Eq. (3.68) and Eq. (3.69a-d).

Table 3.4: Nominal values of the parameters in issue 3

w_P	1	[.]	P_{max}	0.1	[.]
w_R	5	[.]	Q_{max}	2.2	[kJ/h]
k_1	0.001	[L/mol/h]	$u_{A,min}$	0.1	[L/h]
k_2	0.002	[L/mol/h]	$u_{A,max}$	1	[L/h]
$C_{A,in}$	10	[mol/L]	$u_{B,min}$	0.1	[L/h]
$C_{B,in}$	5	[mol/L]	$u_{B,max}$	1	[L/h]
ΔH_1	-10	[kJ/mol]	w	1	[.]
ΔH_2	-10	[kJ/mol]	ε	10	[.]

3.3.4 Concluding remarks

In order to guarantee the feasible input updates during iterative optimization, a new formulation for modifier adaptation scheme is proposed. Instead of zeroth and first-order modifier, high-order convex optimization is employed to update the model where the several coefficients are predefined to guarantee the feasibility. Whereas sufficiently large values of the coefficients allow for the feasible updates, it can make the rate of convergence to the optimum slow. Several theorems about the feasibility, convergence and satisfaction of KKT conditions show that the proposed modifier adaptation can find real plant optimum in spite of model-plant mismatch if the suggested assumptions are satisfied. The illustrative example of optimization for continuous stirred-tank reactor at steady state shows that the proposed method is available and feasible for the multiple constraints case, which generally fails by the conventional modifier adaptation.

Chapter 4

Conclusions and future works

4.1 Conclusions

The modifier adaptation has an advantage of guaranteeing that the converged point satisfies the necessary conditions of optimality or KKT conditions of plant, though there exists model-plant mismatch. While applying the modifier adaptation to the industrial processes, 3 issues are considered and studied in this thesis: repeated large disturbances, efficient estimation of experimental gradient when the number of manipulated variable is large, robust structure of modifier adaptation. Each issue is solved as follows:

First, the problem of performance decrease by repeated large disturbance is solved by applying a feed-forward decision maker based on the historical disturbance data. The feed-forward decision maker is designed by an up-to-date machine learning scheme, deep neural network. According to the simulation results of various data state cases, it is concluded that the feed-forward decision maker provides better starting points and as a result, it improves the performance of the iterative optimization compared to the conventional modifier adaptation.

Second, utilization of the regression methods is proposed for estimating the experimental gradient efficiently. Performing an opti-

mization while estimating the experimental gradient may show a very slow convergence rate, when the number of manipulated variable is large such as the case of run-to-run optimization of fed-batch process. Several regression methods such as multiple linear regression (MLR), principle component analysis (PCA), partial least squares (PLS) are applied to the estimation of gradient and their efficiency are compared. Moreover, a new gradient estimating rule using moving average point is proposed and its satisfaction of plant KKT conditions are proven.

Lastly, a novel structure of modifier adaptation is proposed to handle the absence of prior information about model and the noisy measurement. The proposed scheme approximates the modifying model as a high-order polynomial. Because the approximates is expressed as convex formulation, the model adequacy which is a necessary condition for converge of modifier adaptation is satisfied. In addition, its convergence, satisfaction of plant's KKT condition and robustness under the noisy measurement is proven analytically.

4.2 Future works

The studies in this thesis have improved the performance of the modifier adaptation and have solved realistic problems with industrial chemical and biological processes. Besides these, however, there are still many subjects to improve the performance of the data-driven optimization.

First of all, the deepened subjects for the above 3 issues can be considered. Regarding the first issue of designing the feed-forward decision maker to handle the repeated disturbance, various machine

learning techniques such as convolution neural network (CNN) and recurrent neural network (RNN) can be applied to cope with various disturbance environments. CNN is mainly used for processing multivariate data such as image processing, and RNN is mainly used for processing sequential data such as time series data. In addition, ensemble learning or just-in-time learning coping with changes of disturbance environment such as concept drift can be applied to the design of feed-forward decision maker. Finally, it is also necessary to study the effect of unknown disturbance on process optimization and how to solve it by correlation based modelling such as PCA or PLS, the latent variable space ones. For the second issue of gradient estimation for multivariate systems through regression modelling, more analytical studies on the convergence and noise filtering effects are needed when using the latent variable space model. The factors that can affect convergence to optimal point include the number of principle components and the size of the moving average window. In addition, various regression techniques based on the noise characteristics can be used for experimental gradient estimation. Lastly, for the third issue of the new structural modifier adaptation, the feasibility and convergence should be studied more under the noisy measurement condition. The type of noise whether being uniformly distributed or normally distributed affects the feasibility during the iterative input update and the region of attraction.

Next, apart from the above three issues, the research topic for improving the performance of the overall modifier adaptation can be considered. First, it can be studied the topic of research on RTO in real sense, that is RTO using in-batch measurement, not restricted in the batch-to-batch concept. This is also related to the problem that

modifier adaptation can not be applied properly when frequent disturbances occur within batch units. Although there have been studies about RTO such as modifier adaptation based on transient measurement by François and Bonvin [3], it has obvious limits such as being valid only when the model-plant mismatch is parametric and very small. Thus, efforts should be made to develop a technique for data-based optimization through in-batch measurement using machine learning techniques. Second, the influence of the tuning parameters such as the gain of exponential filter, the coefficients of the new structural modifier adaptation should be studied. It is known that there is trade-off between robustness and convergence rate according to the tuning parameters in the modifier adaptation. Hence, there are needs to study the relationship between the convergence rate and the tuning parameters, especially the new structural modifier adaptation proposed in this thesis.

Bibliography

- [1] François, G., & Bonvin, D. (2013). Use of convex model approximations for real-time optimization via modifier adaptation. *Industrial & Engineering Chemistry Research*, **52**(33), 11614-11625.
- [2] Chachuat, B., Srinivasan, B., & Bonvin, D. (2009). Adaptation strategies for real-time optimization. *Computers & Chemical Engineering*, **33**(10), 1557-1567.
- [3] François, G., & Bonvin, D. (2013). Use of transient measurements for the optimization of steady-state performance via modifier adaptation. *Industrial & Engineering Chemistry Research*, **53**(13), 5148-5159.
- [4] Jia, R., Mao, Z., & Wang, F. (2016). Self-correcting modifier-adaptation strategy for batch-to-batch optimization based on batch-wise unfolded PLS model. *The Canadian Journal of Chemical Engineering*, **94**(9), 1770-1782.
- [5] Marchetti, A., Chachuat, B., & Bonvin, D. (2009). Modifier-adaptation methodology for real-time optimization. *Industrial & Engineering Chemistry Research*, **48**(13), 6022-6033.
- [6] Gao, W., & Engell, S. (2005). Iterative set-point optimization of batch chromatography. *Computers & Chemical Engineering*, **29**(6), 1401-1409.
- [7] Camacho, J., Picó, J., & Ferrer, A. (2007). Self-tuning run to run optimization of fed-batch processes using unfold-PLS. *AIChE journal*, **53**(7), 1789-1804.
- [8] Camacho, J., Lauri, D., Lennox, B., Escabias, M., & Valderrama, M. (2015). Evaluation of smoothing techniques in the run to run optimization of fed-batch processes with u-PLS. *Journal of Chemometrics*, **29**(6), 338-348.

- [9] Jia, R., Mao, Z., Wang, F., & He, D. (2015). Batch-to-batch optimization of cobalt oxalate synthesis process using modifier-adaptation strategy with latent variable model. *Chemometrics and Intelligent Laboratory Systems*, **140**, 73-85.
- [10] Faulwasser, T., & Bonvin, D. (2014). On the use of second-order modifiers for real-time optimization. *IFAC Proceedings Volumes*, **47**(3), 7622-7628.
- [11] Bunin, G. A., François, G., Srinivasan, B., & Bonvin, D. (2011). Input filter design for feasibility in constraint-adaptation schemes. *IFAC Proceedings Volumes*, **44**(1), 5585-5590.
- [12] Shang, C., Yang, F., Huang, D., & Lyu, W. (2014). Data-driven soft sensor development based on deep learning technique. *Journal of Process Control*, **24**(3), 223-233.
- [13] Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, **18**(7), 1527-1554.
- [14] Deng, L., Li, J., Huang, J. T., Yao, K., Yu, D., Seide, F., & Gong, Y. (2013, May). Recent advances in deep learning for speech research at Microsoft. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 8604-8608). IEEE.
- [15] Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2007). Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems*, **19**, 153.
- [16] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, **521**(7553), 436-444.
- [17] Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, **15**(1), 1929-1958.

- [18] Chen, X. W., & Lin, X. (2014). Big data deep learning: challenges and perspectives. *IEEE Access*, **2**, 514-525.
- [19] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 1026-1034).
- [20] Glorot, X., & Bengio, Y. (2010, March). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (pp. 249-256).
- [21] Demuth, H. B., Beale, M. H., De Jess, O., & Hagan, M. T. (2014). *Neural network design*. Martin Hagan.
- [22] Michalski, R. S., Carbonell, J. G., & Mitchell, T. M. (Eds.). (2013). *Machine learning: An artificial intelligence approach*. Springer Science and Business Media.
- [23] Ciregan, D., Meier, U., & Schmidhuber, J. (2012, June). Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on* (pp. 3642-3649). IEEE.
- [24] Sun, Y., Wang, X., & Tang, X. (2014). Deep learning face representation from predicting 10,000 classes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1891-1898).
- [25] Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., & Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, **29**(6), 82-97.
- [26] Collobert, R., & Weston, J. (2008, July). A unified architecture for natural language processing: Deep neural networks with multitask

- learning. In *Proceedings of the 25th international conference on Machine learning* (pp. 160-167). ACM.
- [27] Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)* (pp. 807-814).
- [28] Camacho, J., Lauri, D., Lennox, B., Escabias, M., & Valderrama, M. (2015). Evaluation of smoothing techniques in the run to run optimization of fed-batch processes with u-PLS. *Journal of Chemometrics*, **29**(6), 338-348.
- [29] De Jong, S. (1993). SIMPLS: an alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, **18**(3), 251-263.
- [30] Lopez-Montero, E. B., Wan, J., & Marjanovic, O. (2015). Trajectory tracking of batch product quality using intermittent measurements and moving window estimation. *Journal of Process Control*, **25**, 115-128.
- [31] François, G., Srinivasan, B., & Bonvin, D. (2012). Comparison of six implicit real-time optimization schemes. *Journal Européen des Systèmes Automatisés*, **46**(EPFL-ARTICLE-170545), 291-305.
- [32] Moase, W. H., Manzie, C., & Brear, M. J. (2010). Newton-like extremum-seeking for the control of thermoacoustic instability. *IEEE Transactions on Automatic Control*, **55**(9), 2094-2105.
- [33] François, G., Srinivasan, B., & Bonvin, D. (2005). Use of measurements for enforcing the necessary conditions of optimality in the presence of constraints and uncertainty. *Journal of Process Control*, **15**(6), 701-712.
- [34] Navia, D., Martí, R., Sarabia, D., Gutierrez, G., & de Prada, C. (2012). Handling infeasibilities in dual modifier-adaptation method-

ology for real-time optimization. *IFAC Proceedings Volumes*, **45**(15), 537-542.

- [35] Navia, D., Briceño, L., Gutiérrez, G., & De Prada, C. (2015). Modifier-Adaptation Methodology for Real-Time Optimization Reformulated as a Nested Optimization Problem. *Industrial & Engineering Chemistry Research*, **54**(48), 12054-12071.
- [36] Box, G. E. P., & Draper, N. R. (1969). Evolutionary operation: A statistical method for process improvement.
- [37] Roberts, P. D. (1979). An algorithm for steady-state system optimization and parameter estimation. *International Journal of Systems Science*, **10**(7), 719-734.
- [38] Jäschke, J., & Skogestad, S. (2011). NCO tracking and self-optimizing control in the context of real-time optimization. *Journal of Process Control*, **21**(10), 1407-1416.
- [39] Brdyś, M. A., & Tatjewski, P. (1994). An algorithm for steady-state optimizing dual control of uncertain plants. *IFAC Proceedings Volumes*, **27**(11), 215-220.
- [40] Brdyś, M. A., & Tatjewski, P. (2005). *Iterative algorithms for multi-layer optimizing control*. Imperial College Press, London, UK.
- [41] Gros, S., Srinivasan, B., & Bonvin, D. (2009). Optimizing control based on output feedback. *Computers & Chemical Engineering*, **33**(1), 191-198.
- [42] Marchetti, A., Chachuat, B., & Bonvin, D. (2010). A dual modifier-adaptation approach for real-time optimization. *Journal of Process Control*, **20**(9), 1027-1037.
- [43] Roberts, P. D. (2000). Broyden derivative approximation in ISOPE optimising and optimal control algorithms. *IFAC Proceedings Volumes*, **33**(16), 293-298.

- [44] Huber, P. J. (2011). Robust statistics. In *International Encyclopedia of Statistical Science* (pp. 1248-1251). Springer Berlin Heidelberg.
- [45] Geladi, P., & Kowalski, B. R. (1986). Partial least-squares regression: a tutorial. *Analytica chimica acta*, **185**, 1-17.
- [46] Gao, W., Wenzel, S., and Engell, S. (2015). Comparison of modifier adaptation schemes in real-time optimization. *IFAC-PapersOnLine*, **48**(8), 182-187.
- [47] Gao, W., Wenzel, S., & Engell, S. (2016). A reliable modifier-adaptation strategy for real-time optimization. *Computers & Chemical Engineering*, **91**, 318-328.
- [48] Flores-Cerrillo, J., & MacGregor, J. F. (2005). Latent variable MPC for trajectory tracking in batch processes. *Journal of Process Control*, **15**(6), 651-663.
- [49] Golshan, M., MacGregor, J. F., Bruwer, M. J., & Mhaskar, P. (2010). Latent Variable Model Predictive Control (LV-MPC) for trajectory tracking in batch processes. *Journal of Process Control*, **20**(4), 538-550.
- [50] Lopez-Montero, E. B., Wan, J., & Marjanovic, O. (2015). Trajectory tracking of batch product quality using intermittent measurements and moving window estimation. *Journal of Process Control*, **25**, 115-128.
- [51] Yoo, S. J., Kim, J. H., & Lee, J. M. (2014). Dynamic modelling of mixotrophic microalgal photobioreactor systems with time-varying yield coefficient for the lipid consumption. *Bioresource Technology*, **162**, 228-235.
- [52] Yoo, S. J., Jeong, D. H., Kim, J. H., & Lee, J. M. (2016). Optimization of microalgal photobioreactor system using model predictive control with experimental validation. *Bioprocess and Biosystems engineering*, **39**(8), 1235-1246.

- [53] Orth, J. D., Thiele, I., & Palsson, B. Ø. (2010). What is flux balance analysis?. *Nature Biotechnology*, **28**(3), 245-248.
- [54] Lee, J. M., Gianchandani, E. P., & Papin, J. A. (2006). Flux balance analysis in the era of metabolomics. *Briefings in Bioinformatics*, **7**(2), 140-150.
- [55] Kauffman, K. J., Prakash, P., & Edwards, J. S. (2003). Advances in flux balance analysis. *Current Opinion in Biotechnology*, **14**(5), 491-496.
- [56] Meadows, A. L., Karnik, R., Lam, H., Forestell, S., & Snedecor, B. (2010). Application of dynamic flux balance analysis to an industrial *Escherichia coli* fermentation. *Metabolic Engineering*, **12**(2), 150-160.
- [57] Hanly, T. J., & Henson, M. A. (2011). Dynamic flux balance modeling of microbial co-cultures for efficient batch fermentation of glucose and xylose mixtures. *Biotechnology and Bioengineering*, **108**(2), 376-385.
- [58] Jeong, D. H., Yoo, S. J., Kim, J. H., & Lee, J. M. (2016). Computationally efficient dynamic simulation of cellular kinetics via explicit solution of flux balance analysis: xDFBA modelling and its biochemical process applications. *Chemical Engineering Research and Design*, **113**, 85-95.
- [59] Biegler, L. T., Cervantes, A. M., & Wächter, A. (2002). Advances in simultaneous strategies for dynamic process optimization. *Chemical Engineering Science*, **57**(4), 575-593.
- [60] Biegler, L. T. (2007). An overview of simultaneous strategies for dynamic optimization. *Chemical Engineering and Processing: Process Intensification*, **46**(11), 1043-1053.

- [61] Kameswaran, S., & Biegler, L. T. (2006). Simultaneous dynamic optimization strategies: Recent advances and challenges. *Computers & Chemical Engineering*, **30**(10), 1560-1575.
- [62] Nocedal, J., & Wright, S. J. (2006). *Sequential quadratic programming* (pp. 529-562). Springer New York.
- [63] Gen, M., & Cheng, R. (2000). *Genetic algorithms and engineering optimization* (Vol. 7). John Wiley and Sons.
- [64] Diehl, M., Bock, H. G., Schlöder, J. P., Findeisen, R., Nagy, Z., & Allgöwer, F. (2002). Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control*, **12**(4), 577-585.
- [65] Adetola, V., & Guay, M. (2010). Integration of real-time optimization and model predictive control. *Journal of Process Control*, **20**(2), 125-133.
- [66] Basak, K., Abhilash, K. S., Ganguly, S., & Saraf, D. N. (2002). On-line optimization of a crude distillation unit with constraints on product properties. *Industrial & engineering chemistry research*, **41**(6), 1557-1568.
- [67] Kadam, J. V., Marquardt, W., Schlegel, M., Backx, T., Bosgra, O. H., Brouwer, P. J., & De Wolf, S. (2003). Towards integrated dynamic real-time optimization and control of industrial processes. In *Proceedings foundations of computer-aided process operations (FOCAPO2003)* (pp. 593-596).
- [68] Kadam, J. V., Schlegel, M., Marquardt, W., Tousain, R. L., Van Hessem, D. H., van den Berg, J., & Bosgra, O. H. (2002). A two-level strategy of integrated dynamic optimization and control of industrial processes - a case study. *Computer Aided Chemical Engineering*, **10**, 511-516.

- [69] Hjersted, J. L., & Henson, M. A. (2006). Optimization of Fed-Batch *Saccharomyces cerevisiae* Fermentation Using Dynamic Flux Balance Models. *Biotechnology Progress*, **22**(5), 1239-1248.
- [70] Höffner, K., Harwood, S. M., & Barton, P. I. (2013). A reliable simulator for dynamic flux balance analysis. *Biotechnology and Bioengineering*, **110**(3), 792-802.
- [71] Mahadevan, R., Edwards, J. S., & Doyle, F. J. (2002). Dynamic flux balance analysis of diauxic growth in *Escherichia coli*. *Biophysical Journal*, **83**(3), 1331-1340.
- [72] Schilling, C. H., Edwards, J. S., Letscher, D., & Palsson, B. Ø. (2000). Combining pathway analysis with flux balance analysis for the comprehensive study of metabolic systems. *Biotechnology and Bioengineering*, **71**(4), 286-306.
- [73] Schuetz, R., Kuepfer, L., & Sauer, U. (2007). Systematic evaluation of objective functions for predicting intracellular fluxes in *Escherichia coli*. *Molecular Systems Biology*, **3**(1), 119.
- [74] Varma, A., & Palsson, B. Ø. (1994). Stoichiometric flux balance models quantitatively predict growth and metabolic by-product secretion in wild-type *Escherichia coli* W3110. *Applied and Environmental Microbiology*, **60**(10), 3724-3731.
- [75] Costello, S., François, G., & Bonvin, D. (2016). A directional modifier-adaptation algorithm for real-time optimization. *Journal of Process Control*, **39**, 64-76.
- [76] Marchetti, A. G., Singhal, M., Faulwasser, T., & Bonvin, D. (2017). Modifier adaptation with guaranteed feasibility in the presence of gradient uncertainty. *Computers & Chemical Engineering*, **98**, 61-69.

초 록

개선항 적응법은 데이터 기반 최적화 기법의 일종으로 모델 공정 불일치 조건에서도 수렴하는 값이 공정의 최적 필요 조건을 만족한다는 특징이 있다. 이 학위 논문은 개선항 적응법의 화학 및 생물 공정에 대한 적용 과정에서 발생하는 3 가지 문제점에 대한 해결책을 제시한다. 첫 번째, 반복적으로 발생하는 큰 외란에 의한 최적성 상실의 문제는 과거 외란 정보를 이용하여 앞 먹임 결정기를 디자인 함으로써 빠르게 외란에 대처할 수 있다. 이러한 앞 먹임 결정기는 최신 기법인 심층 신경망 기법을 사용하여 구성하였다. 두 번째, fed-batch reactor 공정의 동적 최적화 문제와 같이 조작 변수의 수가 많은 상황에서 목적 함수와 제약 조건의 실험적 구배를 효과적으로 추정하기 위하여 회귀 분석 방법을 적용하는 방법을 제안하였다. 이를 위하여 multiple linear regression (MLR), principle component analysis (PCA), partial least squares (PLS)와 같은 다양한 회귀 분석 방법이 적용되었고, 보수적인 추정을 위한 moving average 업데이트 방법도 제안되어 수렴했을 때의 공정의 최적 필요 조건 만족이라는 특성을 유지함을 증명하였다. 마지막으로, 업데이트에서 발생할 수 있는 infeasible solution과 공정 노이즈를 효과적으로 처리할 수 있는 새로운 형태의 개선항 적응법을 제안하였다. 또한 제안된 새로운 구조의 개선항 적응법이 갖는 노이즈에 대한 강건성과 수렴성, 그리고 수렴했을 때의 최적 필요조건이 만족함을 증명하였다.

주요어 : 개선항 적응법, 기계 학습, 모델 공정 불일치, 데이터 기반 최적화

학번 : 2012-20974