Korean J. Chem. Eng., **21**(5), 942-949 (2004)

Control of pH Neutralization Process Using Simulation Based Dynamic Programming

Dong Kyu Kim, Kwang Soon Lee* and Dae Ryook Yang†

Department of Chemical and Biological Engineering, Korea University, Seoul 136-701, Korea
*Department of Chemical Engineering, Sogang University, Seoul 121-742, Korea
(*Received 4 May 2004 • accepted 30 June 2004*)

Abstract−The pH neutralization process has long been taken as a representative benchmark problem of nonlinear chemical process control due to its nonlinearity and time-varying nature. For general nonlinear processes, it is difficult to control with a linear model-based control method so nonlinear controls must be considered. Among the numerous approaches suggested, the most rigorous approach is the dynamic optimization. However, as the size of the problem grows, the dynamic programming approach suffers from the curse of dimensionality. In order to avoid this problem, the Neuro-Dynamic Programming (NDP) approach was proposed by Bertsekas and Tsitsiklis [1996]. The NDP approach is to utilize all the data collected to generate an approximation of optimal cost-to-go function which was used to find the optimal input movement in real time control. The approximation could be any type of function such as polynomials, neural networks, etc. In this study, an algorithm using NDP approach was applied to a pH neutralization process to investigate the feasibility of the NDP algorithm and to deepen the understanding of the basic characteristics of this algorithm. As the approximator, the neural network which requires training and the k-nearest neighbor method which requires querying instead of training are investigated. The approximator has to use data from the optimal control strategy. If the optimal control strategy is not readily available, a suboptimal control strategy can be used instead. However, the laborious Bellman iterations are necessary in this case. For pH neutralization process it is rather easy to devise an optimal control strategy. Thus, we used an optimal control strategy and did not perform the Bellman iteration. Also, the effects of constraints on control moves are studied. From the simulations, the NDP method outperforms the conventional PID control.

Key words: pH Neutralization Process, The NDP (Neuro-Dynamic Programming), Constraint on Input Movement, k-Nearest Neighbor Method, Neural Network

INTRODUCTION

Generally, it is often inefficient to control the nonlinear processes with linear control methods. In order to achieve more accurate and precise control performance, the most rigorous solution for the control of nonlinear system is to use the optimal control strategy obtained by dynamic optimization considering the nonlinearity of the process.

The optimal control strategy can be obtained using standard Dynamic Programming (DP). The aim of Dynamic Programming is to find the optimal time-varying input policies by minimizing the objective function which is defined according to the specific control purposes, and in most cases, the optimal strategy is calculated rather numerically than analytically. If the size of problem is large, the calculation load can be enormous and the solution cannot be obtained within the given sampling time even with quite a fast computer. This problem is called as ‘Curse of Dimensionality’ and this makes the on-line control using DP virtually impossible [Kaisare et al., 2003]. However, as the Neuro-Dynamic Programming (NDP) approach is introduced, the application of DP to nonlinear processes becomes possible and the field of application for NDP is growing. This approach is to perform the vast amount of calculation offline, to learn the optimal strategy in a simple form of approximation and to calculate the optimal strategy based on the approximation of cost-to-go function online. Cost-to-go (or profit-to-go) function as a performance objective function can be approximated by a nonlinear function or neural network (NN) and this can reduce the calculation burden so that the dynamic programming approach can be applied online. But the NN requires appropriate training before use and the training of NN is not trivial for many cases. To avoid the difficulty in NN training, local approximation method could be used such as the k-nearest neighbor method (kNN).

In this study, simulation-based DP method suggested by Kaisare et al. [2003] is investigated against a pH neutralization process. Through the simulations, the neural network and the kNN method are compared. An optimal control of pH neutralization process to avoid the Bellman iteration is suggested and the effects of constraints on input moves are investigated.

NEURO DYNAMIC PROGRAMMNG (NDP)

1. Dynamic Programming

A discrete-time dynamic system can be described by an n-dimensional state vector x(k) and an m-dimensional input vector u(k) at time step k. Choice of an m-dimensional control vector u(k) determines the transition of the system from x(k) state to x(k+1) through the following relations [Bertsekas and Tsitsiklis, 1996; Bryson Jr., 1999],

$$x(k+1)=F_h(x(k), u(k)) \tag{1}$$

where F_h denotes the process model equation and h represents the

†To whom correspondence should be addressed.
E-mail: dryang@korea.ac.kr

sampling time. A general dynamic optimization problem for such a system is to find the optimal sequence of control vectors u(k) for k=0, ..., N−1 to minimize a performance index which is related to the cost-to-go function.

Before defining the cost-to-go function, the one-stage-cost, ϕ should be defined. Among many ways, the most popular one-stage-cost can be chosen as follows, with the weighting factors Q and R.

$$\phi(x(k), u(k))=\{Q\times[x(k+1)-x_{sp}]^2\}=\{R\times\Delta u(k)^2\}. \tag{2}$$

where k=0, ..., N−1, $u(0)=u_0$, and x_{sp} denotes the set point.

Then the cost-to-go can be expressed as follow.

$$J_k(x(k))=E\left[\sum_{i=k}^{N-1}\phi(x(i),u(i))+\phi_N\right] \tag{3}$$

where ϕ_N represents the final cost. If N is infinite, then it becomes the infinite horizon cost-to-go function. It can be expressed as a recursive form,

$$J_k(x(k))=E[\phi(x(k), u(k))]+J_{k+1}(F_h(x(k), u(k)) \tag{4}$$

and the above equation can be shown to satisfy the Bellman equation [Bertsekas and Tsitsiklis, 1996].

$$J^*(x(k))=\min_u E[\phi(x(k),u(k))]+J^*(x(k+1)) \tag{5}$$

where E[·] denotes expected value and superscript * implies the optimal value. For simplicity, $J^*(x(k))$ will be shortened as $J^*(k)$. The final goal of DP is to find the input strategy u(k), k=1, ..., N-1 so that the optimal cost-to-go function $J^*(k)$ satisfies the Bellman equation for all time-step k. The solution can usually be obtained numerically and it suffers from the curse of dimensionality when it involves the gridding of large state space dimension. In order to circumvent the problem, one approach suggested by Kaisare et al. [2003] described in the next section can be applied.

2. Simulation-Approximation-Evolution (SAE) Algorithm

The SAE algorithm [Kaisare et al., 2003] is one of the reinforcement learning methods and it involves computation of the converged cost-to-go approximation offline, which is described in Fig. 1. The SAE algorithm is roughly composed of two parts. The first part is "Simulation part." Since the true optimal control strategy is very hard to obtain, the simulation is performed using a suboptimal control law to make training data set which is used for the calculation of the infinite horizon cost-to-go function (Eq. (5)) for each state visited during the simulation. Then the suboptimal cost-to-go function is calculated by

$$J(k)=\sum_{i=k}^{N}\phi(i) \tag{6}$$

where N is sufficiently large for the system to reach a new steady state. The second part is "Cost approximation part". In this part, the cost-to-go function approximation is performed by fitting a neural network or other function approximator to the data from "Simulation Part." In addition to that, Bellman iteration and policy update procedure is performed to improve the approximation of the cost-to-go function if a suboptimal control policy is used [Kaisare et al., 2003; Lee and Lee, 2004].

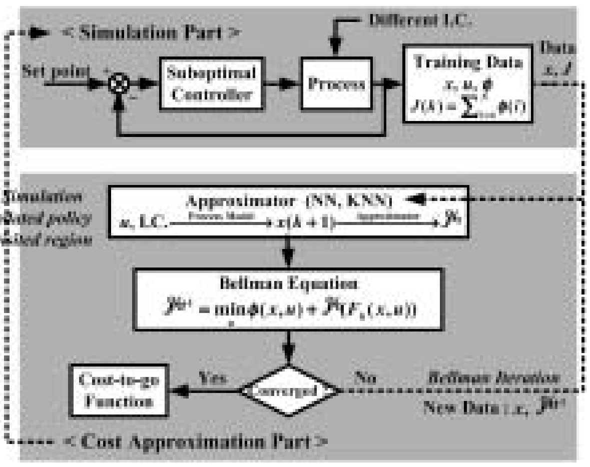


Fig. 1. Architecture for offline computation of cost-to-go approximation.

3. Cost-to-go Approximator

In the algorithms using the neuro-dynamic programming, the performance of the approximator for the cost-to-go approximation is crucial. As approximators, the global approximator and the local approximator can be considered. Global approximators like neural network, polynomial, etc. are the parametric approximators which require extensive offline training, and the local approximators like k-nearest neighbor, kernel-based aproximator, etc. are nonparametric approximators which require extensive querying instead of offline training.

3-1. Neural Network

The neural networks are composed of simple computing elements in parallel. These elements are inspired by biological nervous systems. A neural network (NN) to approximate a particular function can be trained by adjusting the weights of the connections between elements [Demuth and Beale, 1998] as in Fig. 2. Because the NN is one of global approximator, it is difficult to confirm the safeguard against over estimation and the ability of extrapolation even though the computation of function evaluation is easy once trained. Furthermore, the convergence for Bellman iteration using NN is not guaranteed. Thus, the training of NN is quite critical to the performance of the neuro-dynamic programming approaches.

3-2. k-Nearest Neighbor Method

The k-nearest neighbor (kNN) method is a very intuitive method that classifies unlabeled examples based on their similarity with the training set. For a given unlabeled example, $x_u \in \Re^D$ ($\Re^D$ is a

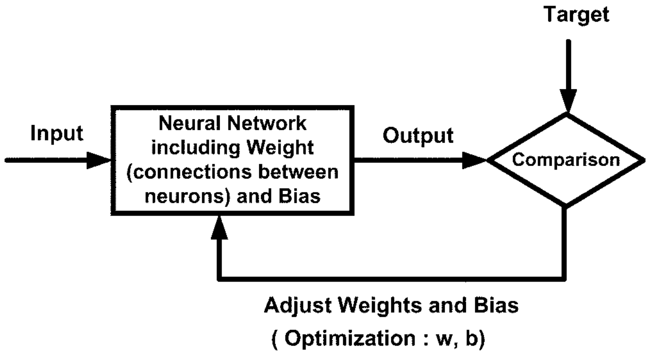


Fig. 2. The schematic diagram for neural network training.

workspace), the k “closest” labeled examples in the training data set are found and assigned as x_u to the class that appears most frequently within the k-subset. The kNN only requires an integer k, a set of labeled examples (training data), and a metric to measure closeness [Osuna, 2002]. The kNN can conveniently handle the quite complex nonlinearity with sufficient data set and training effort is not needed. However, finding the neighboring data set may require extensive data querying procedure. The convergence for Bellman iteration can be guaranteed. But the query time for nearest neighbor is increased in proportion to the number of training data [Lee, 2003].

4. Bellman Iteration

Since the optimal control law is not readily available to begin with, a suboptimal control policy can be used for training of the cost-to-go approximation, and the resulting control law is doomed to be suboptimal. To improve the approximation, the cost or value iteration can be performed until convergence based on the Bellman equation [Kaisare et al., 2003].

$$J^{i+1}=\min_u \phi(x,u)+J^i(F_h(x,u)) \tag{7}$$

This step may impose an enormous computational burden, but it will be performed offline.

pH NEUTRALIZATION PROCESS

The pH neutralization process has long been taken as a representative benchmark problem of nonlinear chemical process control due to its nonlinearity and time-varying nature. In this study, the pH neutralization process is selected as the control target system with neuro-dynamic programming approach.

1. pH Neutralization Process

The neutralization is a chemical reaction. The control objectives are to drive the system to a different pH conditions (tracking control) or to regulate the effluent pH value despite the disturbance by manipulating the flow rate of titrating stream [Henson and Seborg, 1994, 1997]. The process is illustrated in Fig. 3 and the operating conditions are shown in Table 1. The reactor type of the neutralization process is a continuous stirred tank reactor (CSTR) with baffles, which has a volume of 2.5 L. The inlet stream consists of a strong acid stream (q_1: feed solution), a weak acid stream (q_2: buffer solution) and a strong base stream (q_3: titrating solution), which are pumped to the CSTR. It is assumed that the perfect mixing in tank and the complete dissociation in solution at 25 °C occur [Yoo, 2002]. Table 1 shows the typical operating conditions of the process of concern.

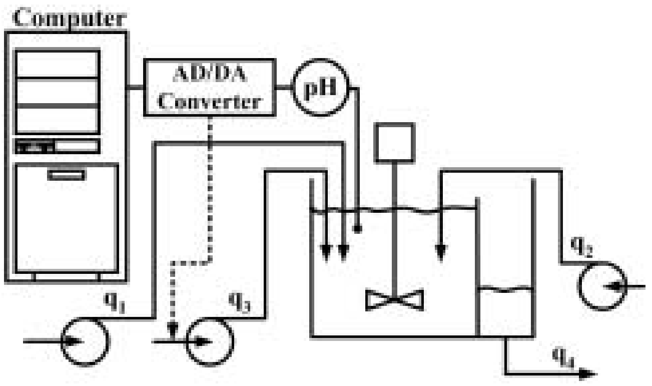


Fig. 3. The pH neutralization process.

Table 1. Operating conditions of pH neutralization process

Symbols	Values	Stream	Composition
V	2,500 [ml]	q_1	0.003 M HNO_3
q_1	9.0 [ml/s]		5.0×10^{-5} M H_2CO_3
q_2	0.6 [ml/s]	q_2	0.01 M $NaHCO_3$
q_3	8.5 [ml/s]	q_3	0.003 M NaOH
			5.0×10^{-5} M $NaHCO_3$

2. pH Neutralization Process Model

Generally, the strong acid-base reaction is always assumed to reach equilibrium in water solution almost instantly. This implies the reaction rates approach infinity. So, the reaction rate terms can be ignored in process model for simplification. Using these assumptions, Gustafsson and Waller [1983] proposed a model using reaction invariants. As the strong acid and base solutions are completely dissociated into ions, the chemical reactions with a weak acid solution reach equilibrium state. The chemical reactions in the system are as follows [Yoo, 2002].

$$\begin{aligned}&H_2CO_3\leftrightarrow HCO_3^-+H^+,\\&HCO_3^-\leftrightarrow CO_3^{2-}+H^+,\\&H_2O\leftrightarrow OH^-+H^+.\end{aligned} \tag{8}$$

The equilibrium constants for the reactions are defined as

$$K_{a1}=\frac{[HCO_3^-][H^+]}{[H_2CO_3]},K_{a2}=\frac{[CO_3^{2-}][H^+]}{[HCO_3^-]},K_w=[H^+][OH^-] \tag{9}$$

The total amount of the reaction invariant is not affected by the degree of chemical reaction. According to this fact, the reaction invariants can be derived from the stoichiometry. As Gustafsson and Waller proposed, two kinds of reaction invariant variables are defined in this process. The first reaction invariant is the concentration of charge related ions, and the other reaction invariant is the total concentrations related to carbonate ions.

$$\begin{aligned}&W_{ai}=[H^+]_i-[OH^-]_i-[HCO_3^-]_i-2[CO_3^{2-}]_i,\\&W_{bi}=[H_2CO_3]_i+[HCO_3^-]_i+[CO_3^{2-}]_i.\end{aligned} \tag{10}$$

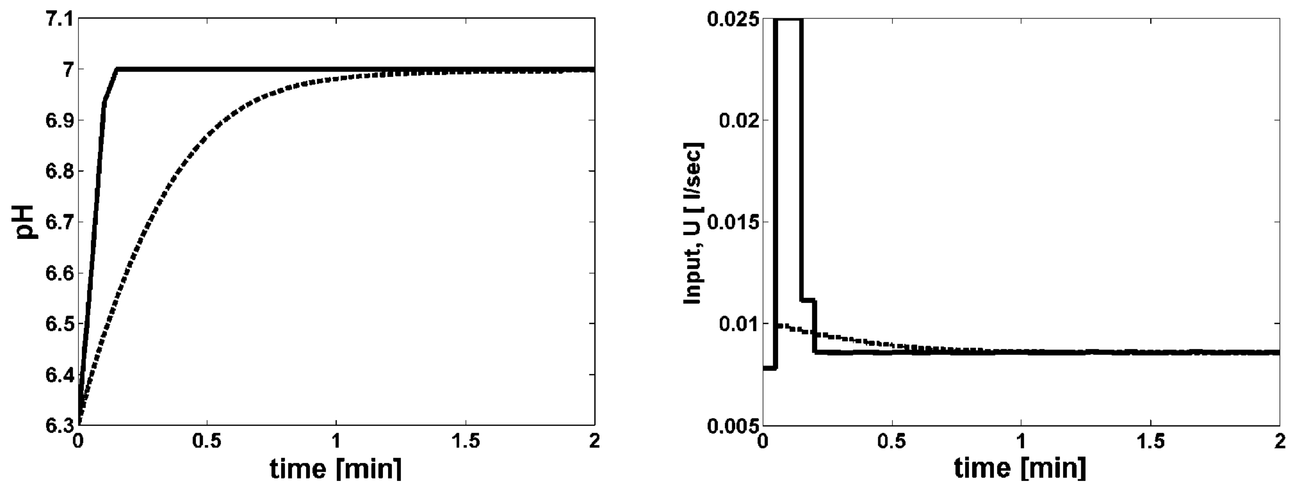
where W_a denotes the charge-related reaction invariant, W_b denotes the carbonate ion-related reaction invariant, and i=1, 2, 3, 4 for each stream in Fig. 3.

The relationship between pH value and the reaction invariants is given by a nonlinear equation from Eqs. (9)-(10) which represent the relation between a hydrogen ion concentration and reaction invariants.

$$W_b\frac{K_{a1}/[H^+]+2K_{a1}K_{a2}/[H^+]^2}{1+K_{a1}/[H^+]+K_{a1}K_{a2}/[H^+]^2}+W_a+\frac{K_w}{[H^+]}-[H^+]=0 \tag{11}$$

The pH value is the negative logarithm of the hydrogen ion concentration (pH=−log[H$^+$]), so the pH value can be determined if W_a and W_b are known.

The dynamic process model for the pH neutralization process can be derived from the component balance for the reaction invariants [Yoo, 2002]:



(a) pH change (b) Input change

Fig. 4. Comparison of results between PI control (········) and NDP approach using NN (——) with respect to a step change in set point (pH 6.3→7).

$$V\frac{dW_{a4}}{dt}=q_1(W_{a1}-W_{a4})+q_2(W_{a2}-W_{a4})+u(W_{a3}-W_{a4})$$
$$V\frac{dW_{b4}}{dt}=q_1(W_{b1}-W_{b4})+q_2(W_{b2}-W_{b4})+u(W_{b3}-W_{b4}) \tag{12}$$

In the above dynamic process model, it is assumed that the flow rates and the concentrations of the feed and buffer streams are known except for two properties, W_{a1} and W_{b2} and they are considered as the unknown parameters (θ). Therefore, the dynamic process model can be obtained as the following state space model [Yoo, 2002]:

$$\dot{x}=f(x, t)+g(x, t)u+F_{\theta}(t)\theta$$
$$c(x, y)=0 \tag{13}$$

where

$$f(x,t)\frac{1}{V}\begin{bmatrix} q_2(W_{a2}-x_1)-q_1x_1 \\ q_1(W_{b1}-x_2)-q_2x_2 \end{bmatrix}, g(x,t)=\frac{1}{V}\begin{bmatrix} W_{a3}-x_1 \\ W_{b3}-x_2 \end{bmatrix},$$

$$F_0(t)=\frac{1}{V}\begin{bmatrix} q_1 & 0 \\ 0 & q_2 \end{bmatrix}, \theta=[W_{a1} \ \ W_{b2}]^T, x=[W_{a4} \ \ W_{b4}]^T,$$

$$u=q_3,\ y=pH_4,\ pK_1=-\log K_{a1},\ pK_2=-\log K_{a2},$$

$$c(x,y)=x_1+10^{y-14}-10^{-y}+x_2\frac{1+2\times 10^{y-pK_2}}{1+10^{pK_1-y}+10^{y-pK_2}}=0.$$

3. Optimal Control Strategy

The suboptimal control law is used in the "Simulation Part" of SAE algorithm. If the suboptimal control is close to optimal control, the improvement of cost-to-go function by Bellman iteration is not necessary. Fortunately, in this process, an optimal control can be devised from a simple principle. The required flow rate of titrating stream to make the mixture of inlet streams with the desired pH value can be calculated from the information of the inlet streams and the additional amount of titrating stream to make the contents of the CSTR at the desired pH value has to be injected in a shortest-possible time. In this manner, the effluent pH value can be reached to the desired value in shortest time without overshoot or undershoot. This control law is not exactly optimal due to the residence time of the effluent stream considering the constraints of the flow rates, but it is close enough to the optimal control law. Moreover, the amount of additional injection of the titrating stream can be adjusted to make the performance better. By using this optimal strategy, the laborious Bellman iteration is omitted in this study.

RESULTS AND DISCUSSIONS

The NDP approach is applied to the pH neutralization process both using a global approximator (NN) and a nonparametric local approximator (kNN).

1. Results using Neural Network

As an approximator, a multilayer feedforward NN is used, which consists of two input states, 5-neuron hidden layer, and 1-neuron output layer. The weighting factors of one-stage-cost function are

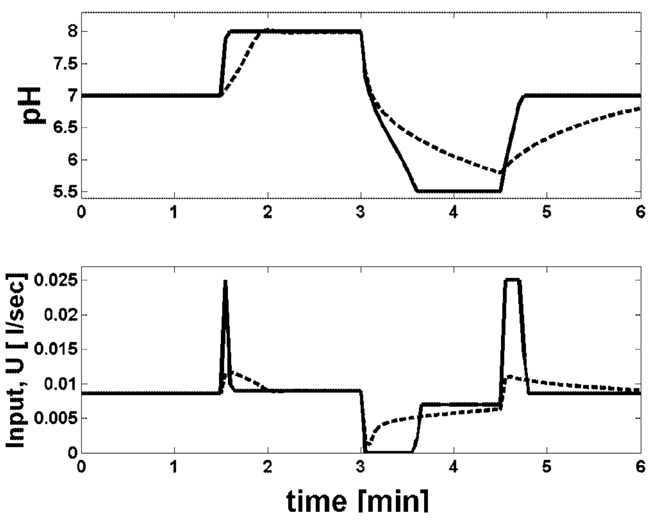


Fig. 5. Comparison of results between PI control (········) and NDP approach using NN (——) with respect to multi-step set point change (pH 7→8→5.5→7).

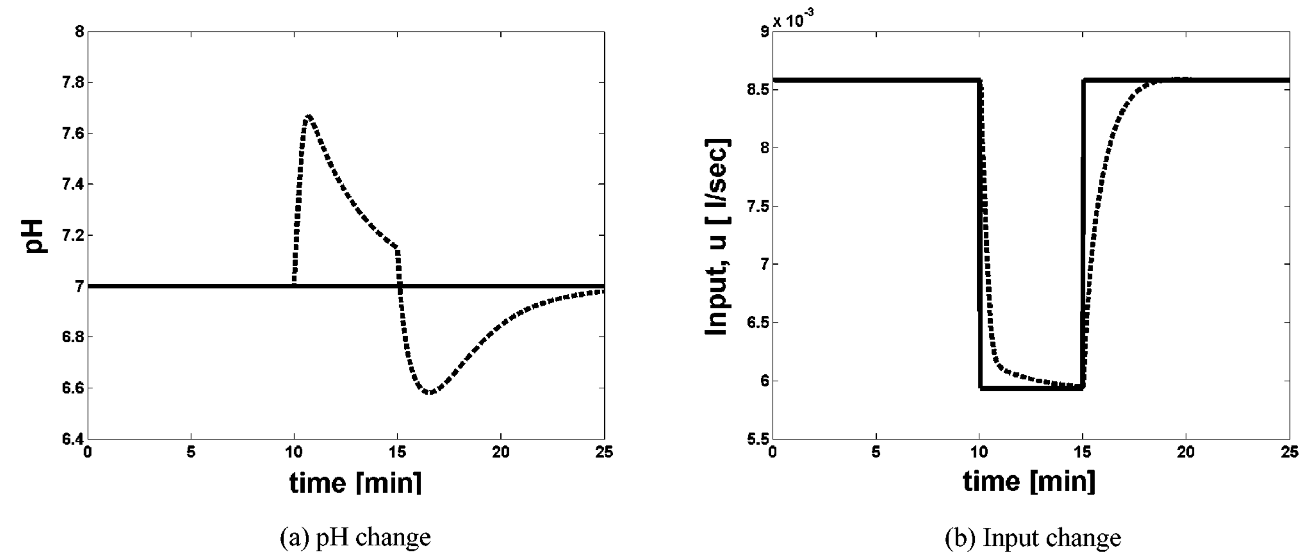


Fig. 6. Comparison of results between PI control (········) and NDP approach using NN (——) with respect to disturbance (15% decreasin W_{a1} at time=10 min).

R=1 and Q=1. For training of NN, the optimal control law described in the previous section is used to generate the training data. The comparisons of the results between well-tuned PI control and NDP for a step change in set point (effluent pH 6.3→7) are shown in Fig. 4. The NDP is much better than PI control as shown Fig. 4. The case of multiple step changes in set point and the disturbance case in feed concentration (W_{a1} change at 10 min) are depicted in Fig. 5 and 6, respectively. From these results, the NDP method outperforms the well-tuned PI control as expected. However, the weighting factor of the one-stage-cost function on error has to be increased to get rid of small steady-state offset. The small steady-state offset in NDP method is caused from the numerical inaccuracy of the NN calculation. In order to overcome this difficulty, either more data around the steady state should be used for training, or the weighting factor for error in one-stage-cost has to be adjusted to emphasize the importance of the error from set point.

2. Results using k-Nearest Neighbor Method

Alternatively, as a local approximator, the kNN method is also applied. The kNN method does not require tedious training as in NN approach and it is very simple to apply. Since the process of our concern is relatively simple, a very good performance of NDP method can be obtained even with only two points nearest neighbor. The performance using kNN method shown in Fig. 7 is almost same as the case using NN. This is because the model we used has only two states and the nonlinearity is not very high. If the process has very complicated nonlinear behavior with many states, the training of neural network is not trivial, and many computational issues regarding training and Bellman iteration can be brought out.

3. Results with Restriction in Δu_{max}

In the cases of previous section, the simulation results are obtained with no restriction in the size of control movement in one sampling time (Δu_{max}) even though there were the lower and upper

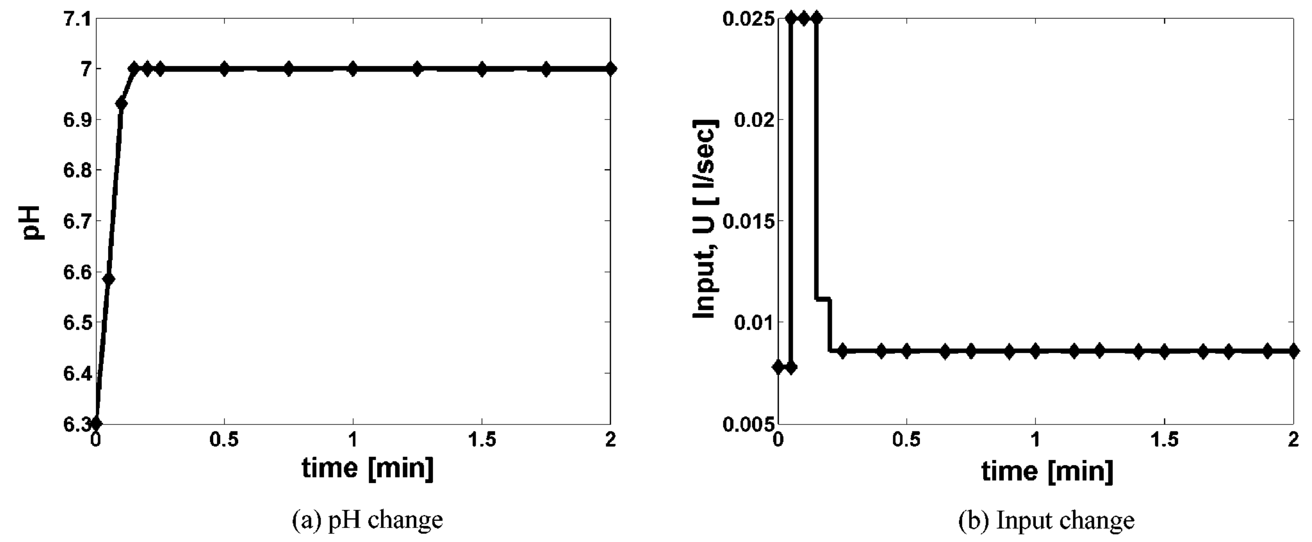
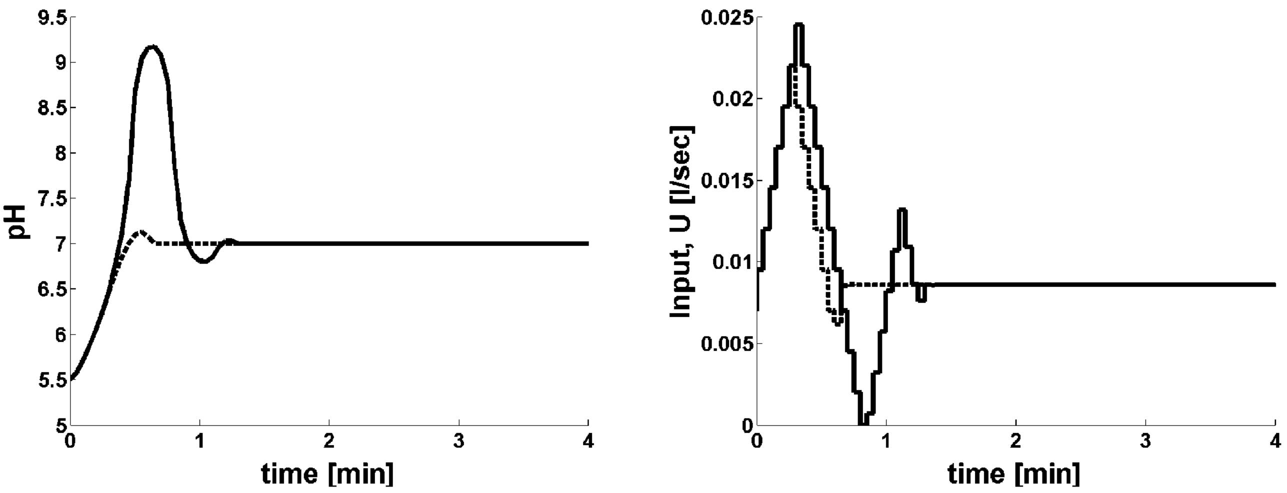


Fig. 7. Comparison of results between NDP approaches using kNN (◆) and NN (——) with respect to set point change (pH 6.3→7).

limits of the control input size (0≤u≤0.025 *l*/sec). But if there is a restriction in Δu_{max}, the NDP control strategy cannot handle the situation correctly. For example, for a step change in set point, NDP method will calculate the additional amount of titrating stream injection without considering the limitation of the control moves in each sampling time and the response may oscillate and the performance will be deteriorated. The NDP controller will increase the control input to inject the needed amount of titrating stream to compensate the difference in the holdups in CSTR assuming that it can stop injecting immediately when needed. However, due to the limitation of control input movement, it cannot decrease the titrating stream to desired level in a sampling time. Thus, it results in over-injection and the process will overshoot and oscillate to compensate the over-injection of the titrating stream. The standard NDP only tries to push the system to the new state as fast as possible under given condition and not to moderate the amount of additional injection considering the limitations and results in overshoot and oscillation. In order to prevent this shortcoming, the standard NDP has to be modified to accommodate the situation. Thus, we suggest that the recursive cost-to-go function calculation should be modified in the following way.

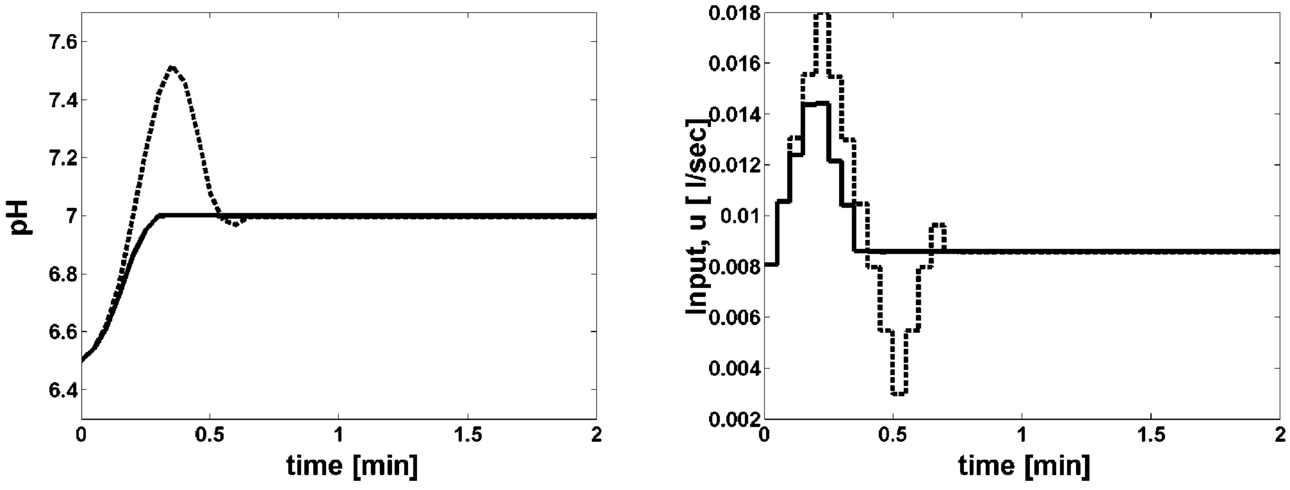
$$J_k^* = \sum_{j=k}^{p+k} \phi_j + J_{k+p}^* \qquad (14)$$

If p=1, Eq. (14) is same as Eq. (5) of original neuro-dynamic approach and if p=∞, it becomes original dynamic programming (DP). This modification increases computational burden to find the optimal input at time step k, but this can prevent the performance degradation due to the constraints on the input change. Fig. 8 shows the performance of the new approach (Δu_{max}=0.0025 *l*/sec) and the overshoot can be reduced significantly in the new approach. Also, the decrease in overshoot for the increase in p is observed. This ap-



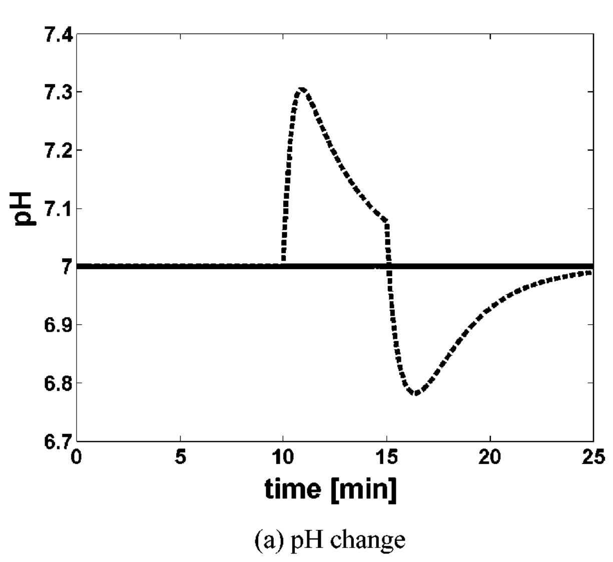
(a) pH change (b) Input change

Fig. 8. Comparison of results from MPC-like NDP approach using kNN with Δu_{max} restriction (in Eq. (14) p=1 (——) and p=4 (⋯⋯)).



(a) pH change (b) Input change

Fig. 9. Comparison of results between standard NDP approach (⋯⋯, 2 states)and NDP approach with augmented states (——, 3 states) when there is Δu_{max} restriction (In the case of NDP approach using kNN).



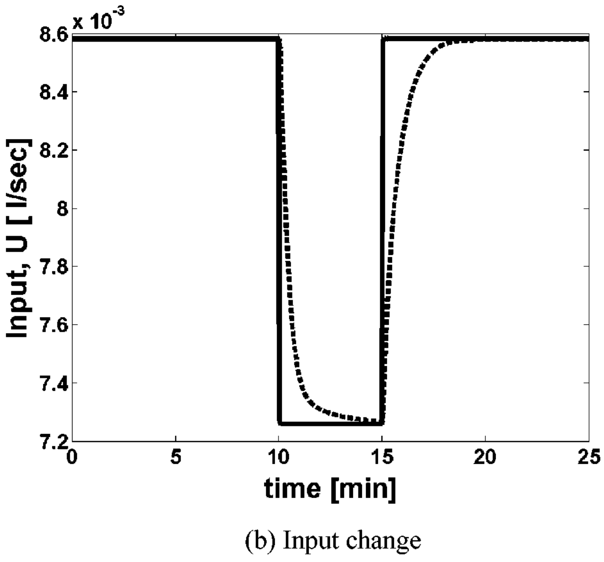


Fig. 10. Comparison of results between PI control (········) and NDP approach using kNN (——) with augmented states with respect to disturbance (15% decrease in W_{a1} at time=10 min).

proach is to incorporate the prediction capability as in Model Predictive Control (MPC) to prevent the performance deterioration due to incorrect information of the process.

Another, yet better approach to resolve this sort of problem is to incorporate the input as a state in the cost-to-go approximation. If we add the information on input to the approximator in NDP algorithm, then the NDP method with augmented states will consider the value of input variable and calculate the required amount of the injection based on the correct information on the process. In Fig. 9, the performance of this approach is shown and the response did not overshoot and provided an excellent performance. In MPC-like approach, the NDP method resulted in a small overshoot even with p=4. However, this approach provided better performance with slight increase in computational burden which is almost negligible. Furthermore, even for the disturbances, the NDP controller with augmented states shows good results as in Fig. 10.

CONCLUSIONS

From the simulation of a pH neutralization process, the NDP method using either the global approximator (NN) or the local approximator (kNN) outperforms the well-tuned PI control. These results are not surprising because NDP method uses much more information and computation. However, if the process is quite complex, this approach can achieve precise optimal control performance without excessive online computational burden. In this study, the NDP approach is applied to a chemical process of pH neutralization and the possibility of applying DP concept even with short sampling period to complex nonlinear chemical processes is verified. In terms of offline preparation of NDP approach, the local approximators such as kNN are preferred over global approximators in the light of cost-to-go approximation. Also, the remedies for the cases of limitation in input movement are suggested.

ACKNOWLEDGEMENT

This work was supported by the Korea Science and Engineering Foundation (R01-2002-000-00574-0).

NOMENCLATURE

E [·]	: expected value
F_h	: process model equation
J	: optimal cost-to-go function
K_{a1}, K_{a2}, K_w	: equilibrium constant in Eq. (9)
Q	: weighting factor of error in the one-stage-cost
R	: weighting factor of input change in the one-stage-cost
V	: reactor volume [ml]
W	: concentration of reaction invariant [M]
h	: sampling time
pH	: pH value for stream
q	: flow rate [ml/sec]
x(k)	: n-dimensional state vector at time step k
x_{sp}	: set point of state
u(k)	: m-dimensional input vector at time step k [*l*/sec]
Δu	: input movement during one sampling time [*l*/sec]
Δu_{max}	: the maximum allowable input movement during one sampling time [*l*/sec]

Greek Letters

ϕ	: one-stage-cost
ϕ_N	: final cost
θ	: unknown parameter

Superscript

*	: optimal value

Subscripts

a	: hydrogen ion related reaction invariant
b	: carbonic ion related reaction invariant
1	: feed stream

2	: buffer stream
3	: base stream
4	: effluent stream

REFERENCES

Bertsekas, D. P. and Tsitsiklis, J. N., "Neuro-Dynamic Programming," Athena Scientific, Massachusetts (1996).

Bryson Jr., A. E., "Dynamic Optimization," Addison-Wesley Longman, Inc., California (1999).

Demuth, H. and Beale, M., "MATLAB Neural Network Toolbox Use's Guide Ver 3.0," The Math Works, MA (1998).

Gustafsson, T. K. and Waller, K. V., "Nonlinear and Adaptive Control of pH," *Ind. Eng. Chem. Res*., **31**, 2681 (1992).

Gutierrez-Osuna, R., "Lecture Note: Introduction to Pattern Recognition," Wright state University (2002).

Henson, M. A. and Seborg, D. E., "Adaptive Input-Output Linearization of a pH Neutralization Process," *Int. J. Adapt. Control Signal Process*, **11**, 171 (1997).

Henson, M. A. and Seborg, D. E., "Adaptive Nonlinear Control of a pH Neutralization Process," *IEEE Trans. on Control Systems Technology*, **2**, 169 (1994).

Kaisare, N. S., Lee, J. M. and Lee, J. H., "Simulation Based Strategy for Nonlinear Optimal Control: Application to a Microbial Cell Reactor," *Int. J. Robust Nonlinear Control*, **13**, 347 (2003).

Lee, J. H., "Lecture Note: From Model Predictive Control to Simulation Based Dynamic Programming: A Paradigm Shift," Georgia Institute of Technology (2003).

Lee, J. M. and Lee, J. H., "Simulation-Based Learning of Cost-To-Go for Control of Nonlinear Processes," *Korean J. Chem. Eng*., **21**, 338 (2004).

Yoo, A., "Experimental Parameter Identification and Control of pH Neutralization Process Based on an Extended Kalman Filter," Master Thesis, Korea University (2002).