# LEARNING CONTROL APPLIED TO pH PLANT

**S. Syafiie*, F. Tadeo*, and E. Martinez****

**Department of Systems Engineering and Automatic Control.*
*University of Valladolid. Science Faculty.*
*Prado de la Magdalena s/n., 47011 Valladolid. Spain.*
*Email: {syam,fernando}@autom.uva.es*
***Consejo Nacional de Investigaciones Científicas y Técnicas,*
*Avellaneda 3657 3000, Santa Fe, Argentina*
*Email: ecmarti@ceride.gov.ar*

Abstract – This paper presents an experimental application of the reinforcement learning method on process control. Reinforcement learning is a model-free technique based on learning how to optimize a cumulative future reward, based on direct experimentation with process plant. As a demonstrative example, reinforcement learning control is tailored and applied on a neutralization laboratory plant: The One-step *Q*-learning rule of reinforcement learning control is applied using a symbolic characterization of plant state. The application shows the ability of the proposed method to learn to control nonlinear chemical processes.

Keywords: learning control, agents, process control, pH control, artificial intelligence.

## 1. INTRODUCTION

The pH control of neutralization processes is a ubiquitous problem encountered in the chemical industry. Traditional linear control analysis and design tools often fail to provide an effective control. Standard PI and PID control often provide poor performance for pH control, due to the high nonlinearity of the chemical/physical systems involved, time varying properties, and the sensibility to small perturbations when working near the equivalence point (Shinskey, 1973; Fuente *et al, 2003*).

Some researchers pay attention to the fine point of methodology for treating pH neutralization process. Different approaches to pH control have been applied previously, for instance fuzzy control (Fuente *et al, 2003*), fuzzy IMC (Edgar and Postlethwaite, 2000), fuzzy predictive control (Biasizzo *et al.*, 1997), and Neural Networks (Loh *et al.*, 1995). Unfortunately, in these approaches there are some weaknesses, such as, the control structures are quite complex (They could be difficult to implement on existing distributed control systems), they are quite conservatives (The controller takes a long time to reach the desired response) and tuning is time-consuming (the controller has many tuning parameters).

For the above reasons, it is necessary to introduce other avenue to solve these problems.

This paper presents a rather simple approach to solve the pH control process problem applying the *Reinforcement Learning* technique. There are several learning algorithms in Reinforcement Learning. In this paper, the one-step ahead *Q*-learning look-up table is applied and the *ε-greedy policy* method is used to select one of the available actions at each state of the plant.

The two main advantages of *Q*-learning are that: 1) it is model-free and 2) it follows an off-policy learning approach. The important advantage of off-policy methods is that for the method sophisticated exploration strategies can be developed to speed up learning.

## 2. REINFORCEMENT LEARNING

In this work, a new technique apply to pH control based on learning is proposed. The use of learning to solve complex control problems is a rather new technique called reinforcement learning or **RL** for

short (Sutton and Barto, 1998; Martínez and Somaglia, 2002). **RL** can be defined as '*learning what to do by doing'*, i.e. how to map perceptions of process states or histories to control actions, so as to maximize an externally provided scalar reward signal. According to this definition, the learner (controller) is not instructed to act under the tutelage of an exemplar teacher, as in most forms of supervised learning, but instead must try control actions seeking out those that provide the maximum cumulative reward.

The learning algorithm in **RL** emphasizes the *interaction* between an active decision-making agent (or intelligent controller) and its target dynamic system (see Figure 1). In the latter, a desired behavior or control *goal* is permanently sought despite imperfect knowledge about system dynamics and the influence of external disturbances, including other controllers. The *reward function* can also incorporate information on one or more *preference* indices. These preferences define the most desirable ways for achieving a control goal (or objective) and are the basis for assigning rewards (or penalties) to a learning controller.

An *agent* (called controller) interacts with its *environments* (controlled system or plant). They interact continually: the agent selects an action and then the environment responds to the action and presents new situation to the agent. These responses of the environment are communicated to the agent through a scalar reinforcement signal. This can be seen in Figure 1.

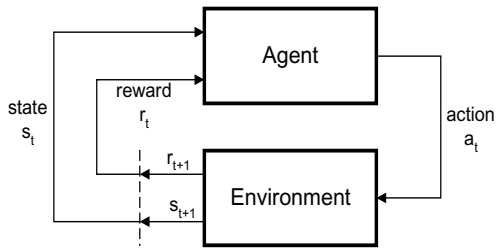


Fig. 1. An agent interacting with its environment. Dash line represents delay.

Achievement of the control goal and preference optimization demands foresight to account for indirect, delayed consequences of control actions. This is particularly critical for chemical plants where material recycles gives rise to process streams whose composition and flow rates continuously evolve and slowly drift over time. This goal-oriented perspective of control actions gives rise to a central problem of **RL**: to devise a computational approach with the capability of apportioning rewards over a sequence of actions, taking into account the control goal to be achieved and the preferences to be optimized. A key concept in this regard is that of an *action-value function*, which has permitted an important breakthrough in the analysis and design of **RL** algorithms (Sutton and Barto, 1998).

When resorting to controller/process interactions for learning, four typical components of an **RL** algorithm can be identified: a *control policy*, a *reward function*, a *value function*, and a *learning algorithm*. The control policy is a dynamic relationship (i.e. changes with interaction) that defines which action to take at a given state bearing in mind the achievement of the goal and optimizing the preferences. The other components serve as means to learn and improve the control law; i.e. their existence is only justifiable on the grounds of being components of learning algorithms for the control law.

When the desired goal has been reached (reward) or when the system has failed (punishment) a signal indicates the success or failure after a sequence of action has been taken. This signal is called *delayed reinforcement signal*.

To maximize the expected value of a criterion function, the delayed reinforcement signal, a learning process of trial and error is applied. These two characteristics (trial-and-error search and delayed reward) are the most important distinguishing features of reinforcement learning (Sutton and Barto, 1998).

### *2.1 Algorithm*

The reward function translates the goal and preferences into single numbers indicating the short-term benefit or reward $r_{t+1}$ of taking the action $a_t$ at a given state $s_t$. Note that the reward is a single number that varies from one decision step to another. However, the very purpose of the control law is to maximize the cumulative reward that can be obtained over the next *h* steps in the sequence of rewards that follow the time step *t* denoted by $r_{t+1}$, $r_{t+2}$, $r_{t+3}$,..., $r_h$. The cumulative reward $R_t$ that can be obtained from time *t* on is just the return, defined as:

$$R_t \equiv r_{t+1}+r_{t+2}+r_{t+3}+...+r_h \qquad (1)$$

To make definition (1) more mathematically tractable for long-sequence of controls (i.e. when $h \rightarrow \infty$) it is better to geometrically discount rewards using a recency factor $\gamma$, (where $0 \leq \gamma \leq 1$):

$$R_t = \sum_{t=0}^{\infty} \gamma^t . r_t \qquad (2)$$

The discount factor $\gamma$ is used to weight near term reinforcements more heavily than distant future reinforcements. If $\gamma$ is small, the agent learns to behave only for short-term reward. The closer $\gamma$ is to 1 the

greater the weight assigned to long-term reinforcements.

Using the return $R_t$, it is possible to assess *how good* (or bad) is to take the action $a_t$ at the state $s_t$, from the point of view of the control goal and the chosen preferences. This forms the basis for defining the action-value function. To clearly distinguish between the effect on $R_t$ of $a_t$ from the effect on $R_t$ of decisions to be taken later in sequel, the action-value function is defined as follows. At time step *t,* the action-value function approximates the expected value of $R_t$ upon executing $a_t$ when $s_t$ is observed and acting *optimally* thereafter. This function is called an *action-value function* and is mathematically defined by

$$Q(s_t, a_t) = E\{R_t | s_t = s, a_t = a\} \quad (3)$$

The main issue to be solved during learning is the dilemma of *exploration* versus *exploitation*. To exploit what it is already known, good estimates of the actual value of actions at different states are needed. For this, exploration of actions with apparently lower values must be tried. To exploit more after learning it is necessary to explore more during learning.

A policy, $\pi(s_t, a_t)$, is defined via the action value function, with represent how much the future rewards the agent would get by taking the action $a_t$ at state $s_t$ and following the current policy in subsequent steps. Equation 3 can be rewritten as an immediate reinforcement, plus a sum of future reinforcements:

$$Q_\pi(s_t, a_t) = E_\pi\left\{R(s_t, a_t) + \sum_{k=1}^{T} \gamma^k R(s_{t+k}, a_{t+k})\right\} \quad (4)$$

The equation can be updated by substituting the sum of future reinforcements with the estimated value function:

$$Q_\pi(s_t,a_t) \leftarrow Q_\pi(s_t,a_t) + E_\pi\{R(s_t,a_t) + \gamma Q_\pi(s_{t+1},a_{t+1}) - Q_\pi(s_t,a_t)\} \quad (5)$$

where the expected discounted reinforcement of taking action $a_t$ is taken over the next state, $s_{t+1}$, given that the current state is $s_t$. A model of state transition probabilities is needed. If it does not exist, a Monte Carlo approach can be used in which a single sample replaces the expectation, and the value function is updated by a fraction of the difference (Anderson *et al*, 1997):

$$Q_\pi(s_t,a_t) \leftarrow Q_\pi(s_t,a_t) + \alpha_t[R(s_t,a_t) + \gamma Q_\pi(s_{t+1},a_{t+1}) - Q_\pi(s_t,a_t)] \quad (6)$$

where the *learning rate*, $0 \le \alpha \le 1$, is tuned to maximize the speed of learning, as small learning rates induce slow learning, and large learning rates induce oscillations. Value iteration is applied to increase the action-selection policy and achieve optimal control. This dynamic programming method combines steps of policy evaluation with policy improvement. The update corresponds to the equation:

$$Q_\pi(s_t,a_t) \leftarrow Q_\pi(s_t,a_t) + \alpha_t\left[R(s_t,a_t) + \gamma \max_{b \in A} Q_\pi(s_{t+1},b) - Q_\pi(s_t,a_t)\right] \quad (7)$$

This equation is known as *Q-learning* algorithm, which is used to improve optimal control.

The balance between *exploration* and *exploitation* can be achieved in different ways. The proposed alternative used in this paper is the ε-greedy approach, which consists of selecting the "best" action most of the time, but every once in a while explore with small probability ε. In the previous work softmax action selection was introduced (Syafiie *et al*, 2004a) and a comparison of softmax and ε-greedy action selection was introduced (Syafiie *et al*, 2004b).

### *2.2 Application to pH process*

This section describes the sequence of steps involved in the implementation of the reinforcement learning approach to the pH process.

To categorize the reading of pH and to select an action available in each state, this study uses 5 symbolic states. These states are called *higher*, *high*, *goal*, *low*, and *lower, and are numerically* represented by state 1, 2, 3, 4, and 5, respectively. Figure 2 shows that the system has 5 states and each state has two possible actions (except in the goal state, where only one action is available). The probability that the process makes a transition to a new state from the current state depends on the system behavior following a control action.. For instance, if the process is in the state high, and the controller chooses action 2, the process may move either to the goal state, to low state, or remain in the state high; but if the controller selects action 3, the process can remain in the state high or may transition to the goal state.

Symbolic states are defined by a parameter that refers to the setpoint, *sp*, as a desired output. The goal state is restricted by boundary values: upper, *sp+0.2,* and lower, *sp-0.2*. This is shown in Figure 3. The goal of the control task is to guarantee that the process is in the goal state most of the time. Maximum reward is get for achieving or remaining in the goal state. When the process makes a transition out of the set point band, the controller is punished by a negative reward.

The process to be studied is the neutralization of a process stream using acid titrated flow rate as the manipulated variable. A selected control action is sent to the system actuator; the process reacts to the action taken and generates a response (state and reward) to the intelligent controller or agent. In the new situation the controller gets a *reinforcement signal* of -1 or 100 depending on the state. This reward expresses the control goal quantitatively. For example, whenever the process state is in lower, low, high or higher states after the control action have been implemented the agent receives a punishment or negative reward of -1. This means that the controller has failed regarding the control goal. Otherwise when the process is in the goal state, the agent receives a reward equal to ± 100. This reward indicates that the agent has been successfully in the selection of the control action.

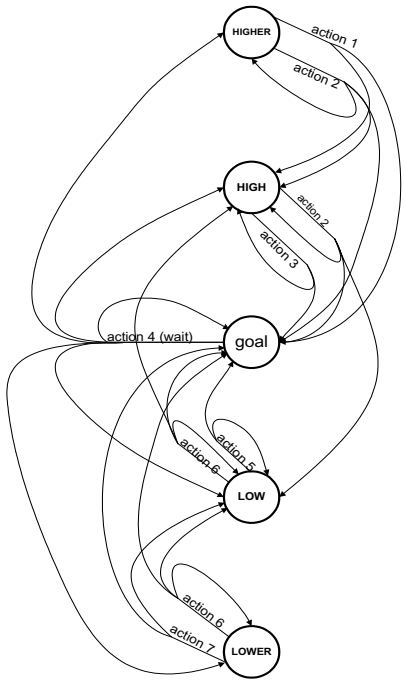


Fig. 2. Action set and possible state action transitions. Each state has 2 actions except for the goal state that has only one. An arrow corresponds to the possibility of transition from current state to a new state.

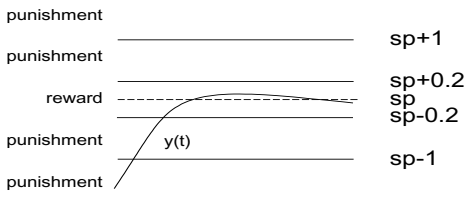


Fig. 3. Defining reward by position of output reading of the process (pH(t)). A reward is defined by applying parameters referring to the setpoint, *sp*, of the process in each state. Each state gives a scalar reward.

A scalar *reinforcement signal* is introduced in equation 8 in order to define rewards following a control action depending on the resulting state. This reward function is defined for each state as a single number.

$$reward = \begin{cases} 100 & if\ \ sp-0.2 < pH < sp+0.2, \\ -1 & if\ \ sp+0.2 < pH < sp+1, \\ -1 & if\ \ sp-1 < pH < sp-0.2, \\ -1 & if\ \ pH > sp+1 \\ -1 & if\ \ pH < sp-1 \end{cases} \quad (8)$$

*Action selection*

When the system is in the goal state, the agent has only one action available, namely to "wait" until the next sampling time. Otherwise, when the system is in any of the other states the *ε-greedy policy* is applied to select one action from the two available. The most probably selected action is the one that has maximum *action-value*. The parameter ε used in the ε-greedy algorithm is 0.3. This means that *exploration* (choosing an action that does not have maximum action-value) will be selected with probability 30 out 100.

The actions 1, 2, 3, 4, 5, 6, and 7 shown in Figure 2 correspond to 0.3, 0.2, 0.1, 0, – 0.1, –0.2 and – 0.3 change percentages, respectively, of the previous control signal $u_{t-1}$. The selected action is then sent to the actuator.

*Control Algorithm*

The algorithm developed for the learning controller is as follow.

1. Read the state $s_t$
2. Select an action $a_t$, this action is chosen from state $s_t$. The action corresponding to maximum value for $s_t$ is chosen with probability (*1-ε*) (unless in goal state that only has one action).
3. Apply the selected $a_t$
4. Receive reward r ($s_t$, $a_t$) and the next state.
5. Find maximum value over the actions for state $s_{t+1}$ as given by max $Q(s_{t+1}, a_{t+1})$ (which is the function that calculates the maximum *Q- value* over the next state, $s_{t+1}$, and current action).
6. Update the *Q-value* for state $s_t$ and action $a_t$, and save it at $Q$ ($s_t$, $a_t$) using Equation (7).

## 3. EXPERIMENTAL SETUP

The pH process that is the focus in this study consists of a tank with a stream of an unknown composition flowing into it. In this study the process is to control the pH level of an aqueous solution of sodium acetate ($NaCH_3COO$) as process flow titrated with hydrochloric acid ($HCl$) in a continuous stirred tank

reactor (CSTR). An over flow system is applied: therefore the volume can be considered constant. Feeding various amounts of a solution of sodium acetate varies the pH solution. The control variable $u$ is the flowrate of the titrating stream, which is fed using a peristaltic pump (ISMATEC MS-1 REGLO/6-160). The output variable $y$ is the concentration of hydrogen ions in the effluent stream, measured as pH. The mixture pH level is measured using an Ag-AgCl electrode (Crison 52-00) and transmitted using a pHmeter (Kent EIL9143). The electrode dynamic response presents appreciable and asymmetric inertia. The pH measure and the control signal are both transmitted through an A/D interface (ComputerBoards CIO-AD16, 0-5V). The plant is controlled and monitored from a Personal Computer. An overview of the process is shown in Figure 4.

The process state at any given time is defined upon pH reading. When the process is in the state 1 or 2, the selected action must be to increase the acid flow into the tank. Otherwise when the process is in either state 4 or 5, the chosen action has to decrease the acid flow into the tank. The agent chooses following a sequence of actions until the goal state is reached.

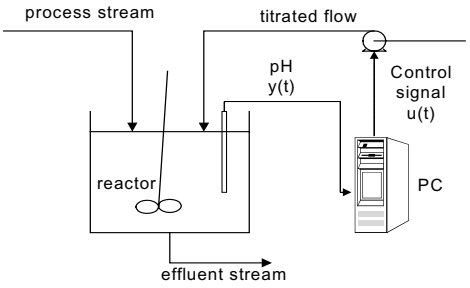


Fig. 4. Experimental setup of pH control process. The objective is to control the pH level of the process stream using acid titrated flow.

## 4. RESULTS AND DISCUSSION

The *Q*-learning algorithm has been successfully applied to the pH process. Different experiments have been carried out to test closed loop performance in realistic situations. The value of the *meta – parameter selected for the agent are:* discount factor (γ) is 0.98, and learning rate (α) is 0.3. The application is conducted on the boundary value of the goal state for reference, sp, (sp - 0.2 < goal < sp + 0.2). This band is selected because it corresponds to the amplitude of the measurement noise. As an example a typical experiment of the response of the plant is shown in figure 5 and control signal is shown in figure 6. When the process is below the setpoint, this means the controller should decrease acid flow into the process, and when the process is above the setpoint the controller must increase the titrated flow into the process. For example, control signal in time $t$-1, $u_{t-1}$, is 0.12 and the system is in state low, the agent takes action 5. The control signal in time t, $u_t$, is 0.11988. The next control signal refers to the control signal $u_t$ and the action taken on given state. The controller manipulates the control signal as follow

$$x = 0.001(a - action)$$
$$u = u + ux \qquad (9)$$

where $a$ is the value for the action corresponding to the goal state that it is defined to be the "wait" action, in this case $a$ is equal to 4.

Figure 5 shows the responses of applying the algorithm for sodium acetate ($NaCH_3COO$) - hydrochloric acid (HCl) system. This experiment is conducted for process flow with unknown concentration of $NaCH_3COO$ titrated with 1 %v HCl. The responses toward the desired setpoint exhibit some oscillations due to the limited number of actions available to the controller when process pH is out of the desired band. The controller, wherever the process is out of the desired band, selects one of only 2 possible actions to manipulate the actuator. These limited actions have been used for control action without prior model of the plant is needed. When the system is within the goal band, the controller only chooses the option of maintaining the titrated flow into the process. The controller can manipulate titrated flow on the range of 0 to 1.

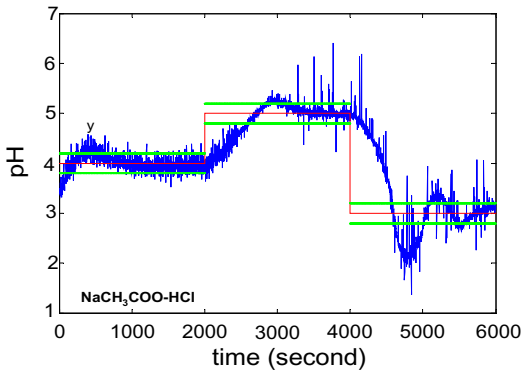


Fig. 5. Step responses of the plant.

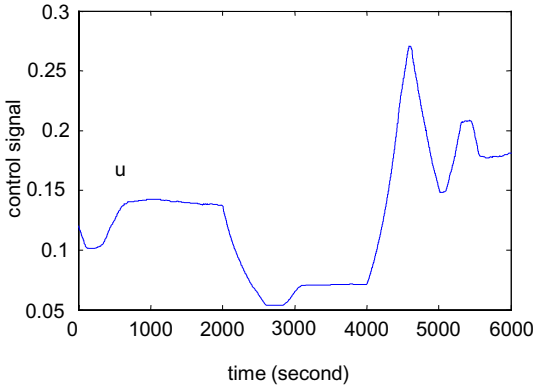


Fig. 6. Learning control signal

When the setpoint changes from 5 to 3, the controller increases acid flow sharply into the tank until the pH has entered the goal band. While the process is within the goal band the agent chooses the previous control signal and the response decreased below the goal band. Soon the agent knows the system outside (below) the goal band, the controller reacts to decrease acid flow into the tank. This is shown in figure 6. The responses after sometime shows that the agent learns to maintain the pH within the desired goal band.

When the environment changes to new situation, in this case when reference is changed, the agent needs sometime to learn and to interact to the new environment. This learning time is called training phase. During the training phase, the controller learns to increase and decrease the titrated flow to maintain the process in the goal band of the reference by adequately increasing and reducing the signals sent to the actuator. This is shown in Figure 6 that the controller is more active when the process is outside the goal band and less aggressive when the pH is within the goal band specification.

## 5. CONCLUSION

The algorithm of one-step ahead *Q*-learning look-up table has been applied in a real plant situation. In this study, the optimal control action is selected using the ε-greedy policy. The application of one-step ahead *Q*-learning look-up table in pH process has given good performance. Difference constituent in solution of acid-base system will give different behavior of the process. Mathematically this will give a different model. The main advantage of the proposed method is that it can be applied to nonlinear process control problems without resorting to a mathematical model of the process.

### *Future work*

In the future work, to solve the oscillation problem, we are planning to apply *multi step actions*, *macro actions,* and *option* for state aggregation and temporal abstraction. Besides pH, Oxidation-Reduction Potential (ORP) is widely used as a control parameter in chemical processes. Other redox systems, ionic strengths of various inorganic salts, polarization of electrodes, organic compounds and temperature effects on electrodes affect the ORP value (Wareham *et al*, 1993). We plan to apply RL for an ORP control. Also we would like to apply RL in an integrated pH and ORP plant.

### *Acknowledgment*

This work was funded by MCYT-CICYT (DPI2003-09392-C02-02) and the first author would like to thank the MAE-AECI for financial support.

## REFERENCES

Anderson, C. W., D. C., Hittle, A. D., Katz, and R. M. Kretchmar (1997), Synthesis of reinforcement learning, neural network, and PI Control Applied to a Simulated Heating Coil, *Journal of Artificial Intelligence in Engineering,* **Vol. 11,** No. 4, pp. 423 – 431

Biasizzo, K. K., I. Skrjanc, and D. Matko (1997), Fuzzy predictive control of highly nonlinearity pH process, *Computer Chemical Engineering*, **Vol. 21,** pp. s613 – s618

Edgar, C. R., and B. E. Postlethwaite (2000), MIMO fuzzy internal model control, *Automatica*, **Vol. 34.** pp. 867 - 877

Fuente, M. J., C. Robles, O. Casado, S. Syafiie, and F. Tadeo (2003), Fuzzy control of neutralization process, *submitted to Chemical Engineering Science*

Loh, A. P., K. O. Looi, and K. F. Fong (1995), Neural network modeling and control strategies for a pH process, *Journal of Process Control*, **Vol. 6.**, pp. 355 - 362

Martínez, E. C. and O. Somaglia, (2002), Reinforcement learning applications to process control and optimization, *Intelligent Systems and Applications*, CRC Press, Leondes, C. (Editor), **Vol. 5.**, pp. 211-238.

Shinskey, F. G. (1973), pH and pIon Control in Process and Waste Stream, Wiley, New York.

Syafiie S., F. Tadeo and E. Martinez (2004a), Learning control application to nonlinear process control, submitted to World Automatic Congress (WAC 2004), Seville, Spain.

Syafiie S., F. Tadeo and E. Martinez (2004b), Softmax and ε-greedy policies applied to process control, submitted to IFAC workshop on Adaptation and Learning in Control and Signal Processing (ALCOSP 2004), Yokohama, Japan

Sutton, R. S., and A. G. Barto (1998), *Reinforcement Learning: an Introduction*, the MIT press, Cambridge, MA

Wareham, D. G., K. J. Hall, and D. S. Mavinic (1993), Real-time control of wastewater treatment system using ORP, Water Science Technology, **Vol. 28.** No. 11, pp 273 – 282.