Extending the Peak Bandwidth of Parameters for Softmax Selection in Reinforcement Learning

Kazunori Iwata, *Member, IEEE*

***Abstract*—Softmax selection is one of the most popular methods for action selection in reinforcement learning. Although various recently proposed methods may be more effective with full parameter tuning, implementing a complicated method that requires the tuning of many parameters can be difficult. Thus, softmax selection is still worth revisiting, considering the cost savings of its implementation and tuning. In fact, this method works adequately in practice with only one parameter appropriately set for the environment. The aim of this paper is to improve the variable setting of this method to extend the bandwidth of good parameters, thereby reducing the cost of implementation and parameter tuning. To achieve this, we take advantage of the asymptotic equipartition property in a Markov decision process to extend the peak bandwidth of softmax selection. Using a variety of episodic tasks, we show that our setting is effective in extending the bandwidth and that it yields a better policy in terms of stability. The bandwidth is quantitatively assessed in a series of statistical tests.**

***Index Terms*—reinforcement learning, softmax selection, parameter bandwidth, asymptotic equipartition property**

I. Introduction

Over the past few decades, agent learning from interaction with an environment has been an important focus of research on artificial intelligence. One of the most successful frameworks for agent learning is reinforcement learning (RL) [1], [2], which has been applied in various fields of engineering [3], [4], [5], [6], [7], [8], [9]. In RL, the sequential decision process of the agent is frequently described as a Markov decision process (MDP). At each time step in the MDP, the agent selects an action according to its policy, and improves the policy using a learning algorithm such as Q-learning or Sarsa [10], [11], [12]. To balance exploratory and exploitative selections, the policy is commonly represented as a probability measure of the actions given a particular state. Exploratory selection means taking an action even though it is unclear whether it is the best action that will provide the maximum return in the future. In contrast, exploitative selection corresponds to selecting an action that the agent estimates to be optimal by that time step. To avoid the risk of failing to notice actions that yield higher returns, the agent needs to make exploratory selections in the early stages of learning, shifting towards exploitative selections later on. The description of the policy is clearly crucial in controlling the shift from exploration to exploitation, and hence, a large number of studies have focused on this aspect (see [13], [14], [15], [16], for example). Here, we neither attempt to give a review of existing studies nor propose a new description of the policy, but rather concentrate on softmax selection, which is one of the most popular policy descriptions in RL. Compared with other methods with multiple parameters (e.g., those described by [13], [14], [15], [16]), softmax selection is simple and easy to implement because there is essentially only one parameter that needs to be tuned. Moreover, softmax selection often works smoothly in controlling the shift in practice if the parameter is appropriately set for the environment (see [17], for example). Therefore, although other methods with multiple well-tuned parameters may be more effective than softmax selection, the latter is still worth revisiting when considering its reduced cost for implementation and parameter tuning. Thus, the ultimate aim of this paper is to minimize the effort involved in tuning the single parameter. To achieve this, we extend the bandwidth around the optimal parameter that yields the maximum return. Our setting for this purpose is related to the asymptotic equipartition property (AEP) of an MDP [18], [19]. The main idea here is to control the number of elements in the typical set, which is defined by the AEP, balancing exploratory and exploitative selections so that most parameters around the optimal parameter remain good. Conducting numerical analysis in episodic tasks, we show that our setting is useful for extending the bandwidth; the results are evaluated quantitatively in a series of statistical tests.

The paper is organized as follows. After reviewing the basics of learning with action-value functions in Section II, we improve the variable setting of softmax selection in Section III. Experimental results are statistically evaluated in Section IV, and our conclusions are presented in Section V.

II. Learning with Action-value Functions

In this section, we explain the basics of RL with action-value functions in an MDP.

A. Markov Decision Process

Let $\mathbb{R}$ and $\mathbb{R}^+$ be real numbers and positive real numbers, respectively, and $\mathbb{N}$ and $\mathbb{N}_0$ be positive integers and nonnegative integers, respectively. We concentrate on episodic tasks formulated by discrete-time MDPs with discrete states, actions, and rewards, unless otherwise noted. $\mathcal{S}$, $\mathcal{A}$, and $\mathcal{R}$ denote, respectively, the non-empty finite sets of states, actions, and rewards, expressed as

$$\mathcal{S} \triangleq \{s_1, s_2, \ldots, s_I\}, \tag{1}$$

$$\mathcal{A} \triangleq \{a_1, a_2, \ldots, a_J\}, \tag{2}$$

$$\mathcal{R} \triangleq \{r_1, r_2, \ldots, r_K\}, \tag{3}$$

where $r_k \in \mathbb{R}$ and $|r_k| < \infty$ for all k. We use t to denote a discrete time step on $\mathbb{N}$. Let $s(t)$, $a(t)$, and $r(t)$

K. Iwata is with the Graduate School of Information Sciences, Hiroshima City University, Hiroshima 731-3194, Japan E-mail: kiwata@hiroshima-cu.ac.jp

Manuscript received March XX, 2015; revised Feb XX, 2016.

be, respectively, a state, action, and reward at time step t. The action-selection probability that the agent takes action a_j in state s_i at time step t is expressed as

$$p_{ij}^{(t)} \triangleq \Pr\left(a(t) = a_j \,|\, s(t) = s_i\right), \tag{4}$$

for all i, j. The state-transition probability from state s_i to state $s_{i'}$ with immediate reward r_k due to action a_j taken at time step t is expressed as

$$p_{iji'k}^{(t)} \triangleq \Pr\left(s(t+1) = s_{i'}, r(t+1) = r_k \,|\, s(t) = s_i, a(t) = a_j\right), \tag{5}$$

for all i, j, i', k. The state-transition probabilities are commonly unknown to the agent. The initial state probability that the agent will begin a new episode in state s_i at time step t is expressed as

$$p_i^{(t)} \triangleq \Pr\left(s(t) = s_i\right), \tag{6}$$

for all i. In each episode, the agent starts from an initial state defined by the initial state probability, and repeats state transitions by taking actions until it reaches a goal state. When an episode is completed on reaching the goal state at time step t, the agent begins the next episode at time step $t+1$ and the number of episodes is incremented.

B. Temporal Difference Learning

Here, we briefly explain Q-learning, which is the most popular method for learning an optimal policy. For all i, j, let $Q_{ij}^{(t)}$ be the action-value function estimate with respect to (s_i, a_j) at time step t. This estimate represents the expected sum of future rewards that will be received after the agent executes action a_j in state s_i at time step t. For each t, the single-time-step observation of $(s(t), a(t), r(t+1), s(t+1)) = (s_i, a_j, r_k, s_{i'})$ updates the action-value function estimate according to

$$Q_{ij}^{(t+1)} = Q_{ij}^{(t)} + \alpha\left(r_k + \gamma \max_{j' \in \mathcal{J}_{i'}} Q_{i'j'}^{(t)} - Q_{ij}^{(t)}\right), \tag{7}$$

where α is the learning rate between 0 and 1, and γ is a discount factor between 0 and 1. $\mathcal{J}_i$ denotes the set of action indices available in state s_i. Under certain conditions [10], [11], for all i, j, the estimate converges to the optimal action-value function defined by

$$Q_{ij}^{*} \triangleq \sum_{i'=1}^{I} \sum_{k=1}^{K} p_{iji'k}\left(r_k + \gamma \max_{j' \in \mathcal{J}_{i'}} Q_{i'j'}^{*}\right). \tag{8}$$

C. Softmax Selection

The agent's policy is generally implemented by describing the action-selection probability in (4). One of the most popular methods for policy implementation is softmax selection [1], [2]. In this method, (4) can be written as

$$p_{ij}^{(t)} = \frac{\exp\left(\beta^{(t)} Q_{ij}^{(t)}\right)}{Z_i\left(\beta^{(t)}\right)}, \tag{9}$$

where $Z_i : \mathbb{R} \to \mathbb{R}$ is the partition function given by

$$Z_i\left(\beta^{(t)}\right) \triangleq \sum_{j' \in \mathcal{J}_i} \exp\left(\beta^{(t)} Q_{ij'}^{(t)}\right). \tag{10}$$

Variable $\beta^{(t)}$ determines how strongly the policy prefers actions with high estimates at time step t. As pointed out in [2], setting the variable requires knowledge of the possible action-value function estimates and their exponents. Since the exponent is sensitive to (rises sharply in response to) an increase in the variable, $\beta^{(t)}$ is carefully increased as t advances, so that the policy tends to be deterministic and optimal.

In general, $\beta^{(t)}$ is set timestep-by-timestep. However, because we are dealing with episodic tasks in this paper, we consider setting $\beta^{(t)}$ at the end of each episode, that is, episode-by-episode, rather than timestep-by-timestep. Such an episode-by-episode setting is convenient for episodic tasks, because the range of the number of episodes is usually known. As an episode-by-episode setting, we focus on the most fundamental form,

$$\beta^{(t)} = cm(t), \tag{11}$$

where $c \geq 0$ is a constant parameter that needs to be tuned, and $m(t) \in \mathbb{N}_0$ denotes the number of times that the agent has reached a goal state, up to and including time step t. It should be noted that $m(t) = 0$ before the agent reaches the goal state for the first time. In fact, $m(t)$ is the number of episodes minus one at time step t. We need to tune the constant parameter to obtain a good result, but tuning it well by trial-and-error is sometimes difficult in practice. One of the main reasons for this is that the allowable bandwidth around the optimal parameter is narrow. Example 1 provides an intuitive example of this, while a more detailed statistical discussion about the peak bandwidth is presented in Section IV.

Example 1 (Peak bandwidth)**.** The shortcut environment [20], [21][1] shown in Fig. 1 consists of $\mathcal{S} = \{S, S1, \ldots, S4, G\}$, $\mathcal{A} = \{a, b\}$, and $\mathcal{R} = \{0, 1\}$. In the figure, each circle represents a state, while a thin arrow exiting a state indicates a state transition. The letter and number associated with each arrow denote, respectively, the action available in the state and the probability of the state transition given by the action. Each thick arrow denotes a scalar reward. In each episode, the agent starts in initial state "S" and repeats state transitions by taking actions according to the policy until it reaches the goal state "G". Each trial consists of 100 episodes. Because each episode yields the same reward regardless of the policy, the objective here is to learn the most efficient policy that allows the agent to reach the goal state as quickly as possible. This is designed by $\gamma < 1$ in Q-learning because the future reward decays with γ. Hence, the optimal policy is to take action a everywhere.

We sampled 101 constant parameters of c at regular intervals from $[0, 1.5]$. Using (11), variable $\beta^{(t)}$ was set to each sampled parameter of c. For each of the sampled parameters, we ran 10,000 trials using Q-learning with $\gamma = 0.9$. We set the learning rate to $\alpha_{ij}(t) = 1/\left(2 + \ln(n_{ij}(t))\right)$, where $n_{ij}(t)$ is the number of times that the agent has taken action $a_j \in \mathcal{A}$ in state $s_i \in \mathcal{S}$ up to and including time step t. In each trial, all the action-value function estimates were initialized as $Q_{ij} = 0$

[1]Since the original environment in [20], [21] is non-episodic, the shortcut environment used in this study has been altered to be episodic.

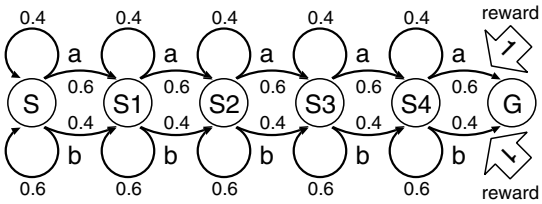


Fig. 1. Shortcut environment.

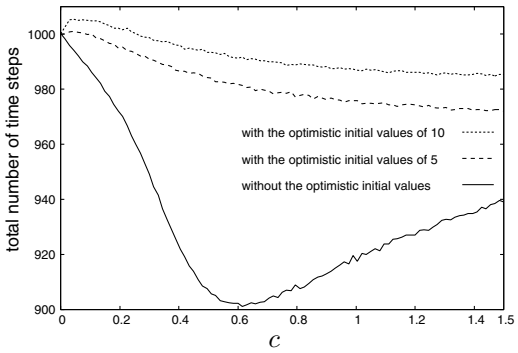


Fig. 2. Efficiency of an agent's policy denoted by the total number of time steps per trial for each $c \in [0, 1.5]$.

for all i, j and policy efficiency was measured as the total number of time steps per trial.

The solid line in Fig. 2 shows the total number of times steps per trial, averaged over 10,000 trials, for each of the sampled parameters. Note that the total number of time steps does not reflect clock time. The smaller the total number of time steps is, the higher is the policy efficiency. We see that c must be around 0.6 and that the bandwidth of good parameters is very narrow, and hence, finding such a parameter will be difficult. Even when dealing with another form of $\beta^{(t)}$, this argument holds in a similar way.

The aim of this paper is simply to extend the bandwidth around the optimal parameter without reducing the performance peak, so that softmax selection can easily be improved. Henceforth, we refer to the bandwidth around the optimal parameter as the peak bandwidth. In Section IV, we make a quantitative assessment of the peak bandwidth by conducting a series of statistical tests.

D. Related Works

Little attention has been paid to how easily optimal tuning of the constant parameter in softmax selection can be done, and hence, there is no directly related work to discuss. As an indirectly related work, we introduce making optimistic initial action-value function estimates. This technique is referred to in [2] as setting optimistic initial values. Setting optimistic initial values is a simple method normally used to incorporate prior knowledge into the policy and/or to encourage exploration. Somewhat interestingly, the optimistic initial values are effective in extending the peak bandwidth in some cases. However, these optimistic initial values have three serious flaws when considering the aim of this paper. First, although learning strongly depends on the optimistic initial values, it is unclear what large values should be used for these. For example, when initializing the action-value function estimates with $Q_{ij}^{(1)} = 5$ and $Q_{ij}^{(1)} = 10$, for all i, j, the resulting total number of time steps per trial depicted by the solid line changed to those depicted by the broken lines in Fig. 2; i.e., the solid line and broken lines in the figure were plotted without and with the optimistic initial values, respectively. Again, note that the total number of time steps does not represent clock time. The smaller the total number of time steps is, the more efficient is the policy. We employed the same settings as those used in Example 1, apart from the initialization, to plot the broken lines. Obviously, 5 or 10 is an optimistic initial value for the shortcut environment in Fig. 1, but it resulted in less desirable policies over the sampling range. Thus, optimistic initial values are not always useful for improving the peak bandwidth, leading us to question what optimistic values are suited to an environment. Second, if there is a loop including a state transition in an MDP, such as taking b in $S2, \ldots, S5$ in Fig. 5 shown later, then the agent tends to repeat its state transition many times until the optimistic initial value on the state transition is reduced to its true value by the updates. This can result in a long computation time. Third, because optimistic initial values are available only in the early learning phase, their use cannot cope with an environment change after this phase. Thus, optimistic initial values do not provide a practical answer.

III. Simple Setting for Extending the Peak Bandwidth

Because our parameter setting method is based on the asymptotic property of stationary ergodic MDPs, we begin this section with a discussion of the asymptotic property. We consider that the asymptotic property is valuable in that it allows us to concentrate on a relatively small set of sequences.

A. Asymptotic Equipartition Property

In RL, we sometimes assume that MDPs are temporarily stationary and ergodic. This assumption not only allows simplicity of analysis but also guarantees that the learning algorithms provide asymptotic convergence to an optimal policy [22], [1], [2]. In stationary ergodic MDPs, $p_{ij}^{(t)}$ and $p_{iji'k}^{(t)}$ are denoted by p_{ij} and $p_{iji'k}$, respectively, because they are not time dependent. This means that the evolution of both $\beta^{(t)}$ and $Q_{ij}^{(t)}$ in (9) is sufficiently slow that $p_{ij}^{(t)}$ can be regarded as being time invariant. Accordingly, in stationary ergodic MDPs, $\beta^{(t)}$ and $Q_{ij}^{(t)}$ are also denoted by β and Q_{ij}, respectively.

We focus on an important property of stationary ergodic MDPs. For all $n \in \mathbb{N}$ and all $x \in \mathcal{X}^n$, let

$$P^n(x) \triangleq \Pr\{(s(1), a(1), s(2), r(2), \ldots, s(n), r(n), a(n), s(n+1), r(n+1)) = x\}, \quad (12)$$

using the convention $r(n+1) = r(1)$, where $\mathcal{X}^n \triangleq \mathcal{S} \times (\mathcal{A} \times \mathcal{S} \times \mathcal{R})^n$. For any $\epsilon \in \mathbb{R}^+$, the typical set with respect

to P^n is defined as

$$C_\epsilon(P^n) \triangleq \{x \in \mathcal{X}^n \mid \exp(-n(H(P)+\epsilon)) \leq P^n(x) \leq \exp(-n(H(P)-\epsilon))\}, \quad (13)$$

where $H(P)$ is called the entropy or stochastic complexity (SC) of the action-selection and state-transition probabilities [18]. Using (4) and (5), $H(P)$ is expressed as

$$H(P) = \left(\sum_{i=1}^{I} v_i \sum_{j=1}^{J} p_{ij} \log \frac{1}{p_{ij}}\right) + \left(\sum_{i=1}^{I}\sum_{j=1}^{J} v_{ij} \sum_{i'=1}^{I}\sum_{k=1}^{K} p_{iji'k} \log \frac{1}{p_{iji'k}}\right), \quad (14)$$

where v_i and v_{ij} denote the stationary distributions

$$v_i \triangleq \Pr\{s(t) = s_i\}, \quad (15)$$

$$v_{ij} \triangleq \Pr\{s(t) = s_i, a(t) = a_j\}. \quad (16)$$

If an MDP is stationary ergodic, then for any $\epsilon \in \mathbb{R}^+$ and all $x \in C_\epsilon(P^n)$,

$$\lim_{n\to\infty} \Pr\left\{\left|-\frac{1}{n}\log P^n(x) - H(P)\right| \geq \epsilon\right\} = 0. \quad (17)$$

For the proof, see [18]. Using $C_\epsilon(P^n)$ in (13), (17) may be rewritten as follows.

1) For all $x \in C_\epsilon(P^n)$,
$$\left|-\frac{1}{n}\log P^n(x) - H(P)\right| \leq \epsilon. \quad (18)$$
2) For sufficiently large n,
$$\Pr\{C_\epsilon(P^n)\} > 1-\epsilon. \quad (19)$$
3) For sufficiently large n,
$$(1-\epsilon)\exp(n(H(P)-\epsilon)) \leq |C_\epsilon(P^n)| \leq \exp(n(H(P)+\epsilon)). \quad (20)$$
Here $|C_\epsilon(P^n)|$ denotes the number of sequences in $C_\epsilon(P^n)$.

The combination of these three items is well known in information theory as the AEP [23]. This property states that all sequences in the typical set have almost the same probability, that the total probability of the sequences is almost one, and that the number of sequences is an exponential function of n determined by the SC. If P^n is not a uniform distribution, then the ratio of the size of the typical set to the total number of sequences is

$$\lim_{n\to\infty} \frac{|C_\epsilon(P^n)|}{|\mathcal{X}^n|} \leq \lim_{n\to\infty} \exp(n(H(P)+\epsilon-\log(IJK))) = 0, \quad (21)$$

which uses the fact that $H(P) < \log(IJK)$ if P^n is not a uniform distribution. Hence, although the typical set is much smaller than $\mathcal{X}^n$, it is important because it has a probability of almost one.

The main idea in this paper is to take advantage of the AEP to extend the peak bandwidth of softmax selection. In terms of probabilistic action-selection such as softmax selection, an explorative policy, which may provide clues for finding a better policy, is to make a random selection that increases the typical set, while an exploitative policy, which may yield a higher return, uses a deterministic selection that reduces the typical set. Controlling the number of sequences in the typical set corresponds to balancing random and deterministic selections so that most parameters around the optimal parameter remain good. It is thus crucial to know how the SC fluctuates as β increases. Substituting (9) for p_{ij} in (14) gives the SC as a function of β. It has been proved in [18] that the SC for stationary ergodic MDPs decreases as β increases. Its derivative is written as

$$\frac{dH(P)}{d\beta} = -\sum_{i=1}^{I} \frac{v_i \beta}{2Z_i(\beta)^2} \times \sum_{j\in\mathcal{J}_i}\sum_{j'\in\mathcal{J}_i} (Q_{ij}-Q_{ij'})^2 \exp(\beta(Q_{ij}+Q_{ij'})). \quad (22)$$

For reference purposes, the proof is given in Appendix A. Intuitively, this derivative quantifies the sensitivity of the SC with respect to β, which is important because the SC determines the number of sequences in the typical set, as shown in the AEP. We see from (22) that the SC depends strongly on v_i and Q_{ij}. However, the conventional form of $\beta^{(t)}$ in (11) does not consider v_i and Q_{ij}. This is one of the main reasons that the peak bandwidth is narrow.

Incidentally, the derivative of the stochastic complexity is derived from the assumption that MDPs are stationary and ergodic. Using the derivative as it is may not be useful for setting β owing to the discrepancy between the assumption and the actual properties in MDPs. Accordingly, we measure the derivative at the end of every episode, and in fact, we use the difference between the measurements in the previous and current episodes. This corresponds to the increment/decrement for b that appears in Algorithm 1.

B. Proposed Setting Method

In this section, we improve the variable setting of softmax selection using the derivative of the SC. In episode-by-episode setting, our setting of $\beta^{(t)}$ is defined by

$$\beta^{(t)} = c(m(t) + b(m(t))), \quad (23)$$

where $c \geq 0$ is a constant parameter. For any $m \in \mathbb{N}_0$, function $b : \mathbb{N}_0 \to \mathbb{Z}$ is defined as

$$b(m) \triangleq |\{m' \mid \zeta(\bar{t}(m'-1)) \leq \zeta(\bar{t}(m')), 2 \leq m' \leq m\}| - |\{m' \mid \zeta(\bar{t}(m'-1)) > \zeta(\bar{t}(m')), 2 \leq m' \leq m\}|, \quad (24)$$

where $\bar{t}(m)$ denotes the first time step in which the agent reaches the goal m times; i.e., this is the time step at the end of the $(m-1)$-th episode. Function $\zeta : \mathbb{N} \to \mathbb{R}$ denotes the

approximate derivative of the SC at time step $\bar{t} \in \mathbb{N}$, given by

$$\zeta\left(\bar{t}\right) \triangleq -\sum_{i=1}^{I} \frac{\hat{v}_i^{(\bar{t})} \beta^{(\bar{t})}}{2Z_i \left(\beta^{(\bar{t})}\right)^2} \times \sum_{j \in \mathcal{J}_i} \sum_{j' \in \mathcal{J}_i} \left(Q_{ij}^{(\bar{t})} - Q_{ij'}^{(\bar{t})}\right)^2 \exp\left(\beta^{(\bar{t})} \left(Q_{ij}^{(\bar{t})} + Q_{ij'}^{(\bar{t})}\right)\right), \tag{25}$$

where $\hat{v}_i^{(\bar{t})} \triangleq n_i\left(\bar{t}\right)/\bar{t}$. $n_i\left(\bar{t}\right)$ is the number of times that the agent has visited state $s_i \in \mathcal{S}$ up to and including time step $\bar{t}$. Note that setting $b(m) = 0$ in (23) provides the conventional setting in (11). For convenience, we use the term "conventional" here to indicate settings without such additional information. There are two reasons for using integer $m(t) + b\left(m(t)\right)$ instead of $m(t)$ in the original form in (11). The first is to align the performance peak on c roughly with that of the original form. For instance, if we used another form, such as $cm(t)b\left(m(t)\right)$, it would be difficult to find certain knowledge about c because the role of c in such a form is distinct from that in the original form. The other reason is so as not to lose the meaning of "time steps" by keeping $b\left(m(t)\right)$ an integer. Function b in (24) may appear complicated, but in fact, its implementation is fairly simple. As shown in Algorithm 1, computing $b(m)$ involves nothing more than incrementing or decrementing it at the end of every episode. Note that m, b, and "prev ζ" in the algorithm denote $m(t)$, $b(m)$, and $\zeta(t)$, respectively. The difference in time complexity between conventional and our settings comes down to the $\zeta(t)$ calculated at the end of every episode, and according to (25), it is $O\left(IJ^2\right)$ per episode. The proposed setting for $\beta^{(t)}$ has the same single parameter. Hence, compared with some action-selection methods with many sensitive parameters discussed in the literature, softmax selection based on the new setting is simple and free of tuning so many sensitive parameters.

C. Role of Function b

Figure 3 plots the evolution of $\zeta(\bar{t}(m))$ derived from the conventional setting,

$$\beta^{(t)} = 0.6m(t). \tag{26}$$

The x-axis in the figure denotes the episode number minus one, which is the number of times that the agent has reached the goal state. We tested the conventional setting with the shortcut environment in Fig. 1. The same settings as those used in Example 1 were employed for the Q-learning. In each trial, all the action-value function estimates were initialized as $Q_{ij}^{(1)} = 0$ for all i, j. Clearly, this meant $\zeta(1) = 0$.

Next, we explain the role of function b using Fig. 3. If $Q_{ij}^{(t)}$ is time invariant for all i, j, then for sufficiently large β, the derivative in (22) is monotonically increasing with respect to β, and tends to zero. Note that "$Q_{ij}^{(t)}$ is time invariant" implies that $Q_{ij}^{(t)}$ has converged. This is the case in Fig. 3, which shows a marked increase in $\zeta(\bar{t}(m))$ for $m \geq 15$. However, as shown by the marked decrease in $\zeta(\bar{t}(m))$ for $m < 15$, in fact, the derivative decreases until $Q_{ij}^{(t)}$ becomes almost time invariant. This is because the Q-learning update for $Q_{ij}^{(t)}$ leads to an increase in $\left(Q_{ij}^{(\bar{t})} - Q_{ij'}^{(\bar{t})}\right)^2$ of the derivative during the initial phase of the trial. In summary, the derivative tends to decrease until all the action-value function estimates converge to their expected values, and thereafter starts increasing. By taking into account this derivative property, function b gives an effective setting for $\beta^{(t)}$. Our setting, defined in (23), automatically controls the increase in the variable by a decrement in b (see Algorithm 1) while the derivative is decreasing, so that the agent frequently adopts an explorative policy in the initial phase, keeping the typical set large. Conversely, our setting encourages the variable to increase by incrementing b while the derivative is increasing, causing the agent frequently to choose an exploitative policy, thereby keeping the typical set small.

Algorithm 1 Computing $b(m)$

```
t ← 1
m ← 0
b ← 0
prev ζ ← 0
for all m such that 0 ≤ m ≤ the number of episodes in each trial do
  s(t) ← an initial state
  while s(t) is not a goal state do
    select an action a(t) according to the policy (variable b is referred to at this point)
    t ← t + 1
    r(t) ← a reward given by the environment
    s(t) ← the next state given by the environment
  end while
  calculate ζ(t)
  if m ≥ 1 and prev ζ ≤ ζ(t) then
    b ← b + 1
  else if m ≥ 1 and prev ζ > ζ(t) then
    b ← b − 1
  end if
  prev ζ ← ζ(t)
end for
```

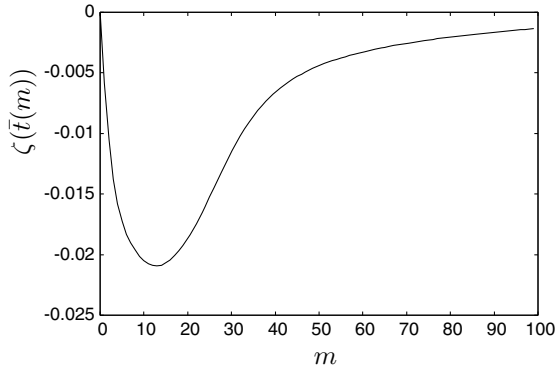


Fig. 3. Episode evolution of the SC derivative with $c = 0.6$.

IV. Experiments

In this section, using a variety of different tasks, we compare our setting in softmax selection to the conventional setting

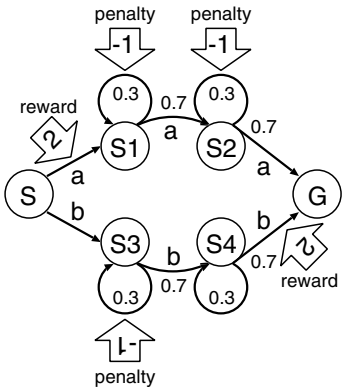


Fig. 4. Misleading environment.

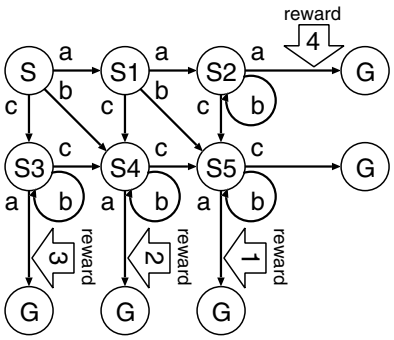


Fig. 5. Deterministic environment.

in terms of policy efficiency and stability. The results are carefully evaluated through multiple statistical comparisons.

A. Environments

We used six episodic tasks provided by the shortcut, misleading, deterministic, entwined, tic-tac-toe, and mountain–car environments. The first four environments are illustrated in Figs. 1, 4, 5, and 6, respectively. In these figures, circles represent states, while a thin arrow exiting a state indicates a state transition. The letter and number associated with each arrow denote an action available in the state and the probability of the state transition given by the action, respectively. Each thick arrow denotes a scalar reward or penalty. In each episode, the agent starts in the initial state "S" and repeats state transitions until it reaches the goal state "G". In each environment, we used Q-learning to estimate the action-value functions. We set the learning rate to $\alpha_{ij}(t) = 1/\left(2 + \ln(n_{ij}(t))\right)$. For each trial, all the action-value functions were initialized as $Q_{ij}^{(1)} = 0$ for all i, j.

a) Shortcut Environment: The shortcut environment, shown in Fig. 1, was used in Example 1. Each trial consisted of 100 episodes. We sampled 101 constant parameters at regular intervals from $[0, 1.5]$. This sampling range was determined to include the peak bandwidths for both the conventional and proposed variable settings. For each of the sampled parameters, we ran 10,000 trials using each variable setting and the Q-learning algorithm with $\gamma = 0.9$. Policy efficiency was measured as the total number of time steps per trial, averaged over 10,000 trials. Note that the smaller the total number of time steps is, the higher is the policy efficiency.

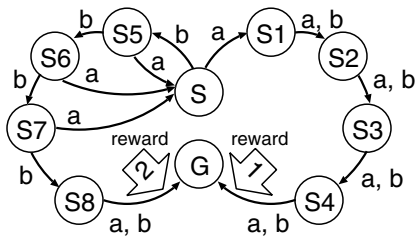


Fig. 6. Entwined environment.

b) Misleading Environment: This environment, illustrated in Fig. 4, consists of $\mathcal{S} = \{S, S1, \ldots, S4, G\}$, $\mathcal{A} = \{a, b\}$, and $\mathcal{R} = \{-1, 0, 2\}$. At first glance, action a in "S" appears better because it gives an immediate reward at the start of the episode. However, the optimal policy in "S" is to take action b. The agent needs to avoid the misleading policy of taking action a in the initial state. Each trial consisted of 100 episodes.

We sampled 101 constant parameters at regular intervals from $[0, 0.5]$, which was determined to include the peak bandwidths for both the conventional and proposed variable settings. For each of the sampled parameters, we ran 10,000 trials using each variable setting and the Q-learning algorithm with $\gamma = 1$. Policy efficiency was measured as the total reward per trial, averaged over 10,000 trials. The larger the total reward is, the higher is the policy efficiency.

c) Deterministic Environment: The deterministic environment, illustrated in Fig. 5, consists of $\mathcal{S} = \{S, S1, \ldots, S5, G\}$, $\mathcal{A} = \{a, b, c\}$, and $\mathcal{R} = \{0, 1, 2, 3, 4\}$. All state transitions here are deterministic. There are five goals with different rewards and three actions in each state. The agent needs to find the goal with the highest reward as efficiently as possible. The optimal policy is always to take action a. Each trial consisted of 400 episodes.

We sampled 101 constant parameters at regular intervals from $[0, 0.1]$, which was determined to include the peak bandwidths for both the conventional and proposed variable settings. For each of the sampled parameters, we ran 10,000 trials using each variable setting and the Q-learning algorithm with $\gamma = 0.99$. Policy efficiency was measured as the total reward per trial, averaged over 10,000 trials. The larger the total reward is, the higher is the policy efficiency.

d) Entwined Environment: The entwined environment [24], [21], [13],[2], shown in Fig. 6, consists of $\mathcal{S} = \{S, S1, \ldots, S8, G\}$, $\mathcal{A} = \{a, b\}$, and $\mathcal{R} = \{0, 1, 2\}$. All state transitions are once again deterministic. The agent needs to perform careful exploration to notice that taking four consecutive b actions from "S" is the optimal policy. Each trial consisted of 400 episodes.

We sampled 101 constant parameters at regular intervals from $[0, 0.1]$, which was determined to include the peak bandwidths of both the conventional and proposed variable settings. For each of the sampled parameters, we ran 10,000

[2]Since the original environment in [24], [21], [13] is non-episodic, the entwined environment here was altered to be episodic.

trials using each variable setting and the Q-learning algorithm with $\gamma = 1$. Policy efficiency was measured as the total reward per trial, averaged over 10,000 trials. The larger the total reward is, the higher is the policy efficiency.

e) Tic-tac-toe Environment: The tic-tac-toe game [1], [2] in which two players, an agent and an opponent, take turns placing their respective marks on a 3×3 board, can be formulated as an episodic task. In the experiment, the agent played a number of games against the opponent, with each game corresponding to one episode. Each trial consisted of 15,000 episodes. In the game, the first player to place three marks in a horizontal, vertical, or diagonal line wins the game. If the board fills up with marks and neither player has three marks in a line, the game ends in a draw. An agent receives four, two, or zero rewards when it wins, draws, or loses the game, respectively. For simplicity, each game began with the opponent placing a mark first. Because a perfect opponent can play so as not to lose, we assume that the opponent chooses an open space with uniform probability in each of its turns. This implies that a game here has nine initial states with the same probability and thus can be described as an MDP. In fact, the MDP has 7536 state-action pairs, which is somewhat larger than the previous environments used. A state-action pair is referred to simply as a state-action. Because so many state-actions require a vast number of observations in Q-learning, we made the most of the observations by considering rotations and reflections of the board. Concretely, whenever an agent observes a subsequent state and reward after executing an action in a state, it is faced with a total of eight possible state-actions related to the state-action taken. The action-value function estimates are subsequently updated according to (7). Because a state-action corresponds to a disposition of marks on the board, the eight state-actions related to the respective state-action represent its rotations of 0, 90, 180, and 270 degrees and its reflections with respect to the vertical, horizontal, and both diagonal axes of the board. Appendix B gives further details of this.

We sampled 101 constant parameters at regular intervals from $[0, 0.005]$, which was determined to include the peak bandwidths of the conventional and proposed variable settings. For each of the sampled parameters, we ran 100 trials using each variable setting and the Q-learning algorithm with $\gamma = 1$. Policy efficiency was measured as the total reward per trial, averaged over 100 trials. The larger the total reward is, the higher is the policy efficiency.

f) Mountain–Car Environment: The mountain–car task [2], in which an underpowered car, regarded as an agent, drives up a steep mountain slope, can be formulated as an episodic task. The agent starts at the bottom of a ravine formed by slopes, and climbs toward the goal at the top of the hill in 1000 time steps (see Fig. 7). The agent moves toward the goal using three possible actions: full-throttle forward, full-throttle reverse, and zero throttle. The state of the agent is represented by a position and velocity, each of which is a real number. For each time step t, the position $y(t+1)$ and velocity $\dot{y}(t+1)$

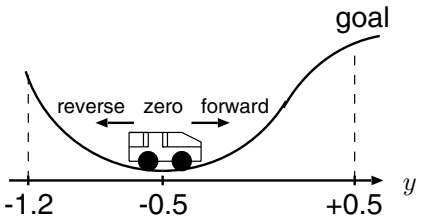


Fig. 7. Mountain–car environment.

are drawn according to a deterministic physical rule:

$$y(t+1) = \text{bound}\left[y(t) + \dot{y}(t+1)\right], \tag{27}$$

$$\dot{y}(t+1) = \text{bound}\left[\dot{y}(t) + 0.001a(t) - 0.0025\cos\left(3y(t)\right)\right], \tag{28}$$

where the bound operation forces $s(t+1)$ and $\dot{s}(t+1)$ into $[-1.2, +0.5]$ and $[-0.07, +0.07]$, respectively. Here, action $a(t)$ is $+1, -1$, and 0 if the agent chooses full-throttle forward, full-throttle reverse, and zero throttle, respectively, at time step t. Whenever the agent's position is less than -1.2, the agent's velocity is reset to zero. A value greater than 0.5 for the agent's position indicates that the agent has reached its goal. The only solution here is to go up the left slope and then drive up the right slope toward the goal using the momentum obtained by going down the slope. Each episode ended with the allocation of 10 rewards when the agent reaches the goal, or no reward if the agent fails to reach the goal after 1000 time steps[3]. Each trial consisted of 200 episodes.

This environment differs substantially from the others in that its state set is not a discrete set. Accordingly, we needed to rewrite the action-value function estimate and the approximate derivative of the SC, to ensure that these were suitable for continuous states. For any position y and velocity $\dot{y}$, the action-value function estimate of action a_j at time step t is expressed as

$$Q_j^{(t)}\left(y, \dot{y}\right) \triangleq \sum_{l=1}^{L} w_{jl}^{(t)} \phi_l\left(y, \dot{y}\right), \tag{29}$$

where $\phi_l : \mathbb{R}^2 \to \mathbb{R}$ denotes the l-th Gaussian basis function defined by

$$\phi_l\left(y, \dot{y}\right) \triangleq \exp\left\{-\frac{\left(y-\mu_l\right)^2}{2\sigma_l^2} - \frac{\left(\dot{y}-\kappa_l\right)^2}{2\rho_l^2}\right\}, \tag{30}$$

and $w_{jl}^{(t)}$ denotes the l-th weight for action a_j. This is a simple linear-function approximation of the action-value function estimate, which is well known in RL [2], [25]. The center of the l-th Gaussian basis function is specified by (μ_l, κ_l). We generated $56\,(=\ 8 \times 7)$ Gaussian basis functions for each action (i.e., $L = 56$) and then placed their centers uniformly across the two-dimensional space created by $\{(y, \dot{y}) \mid y \in [-1.2, +0.5], \dot{y} \in [-0.07, +0.07]\}$. As for the width of the l-th Gaussian basis function, we set $\sigma_l = 0.1$ and $\rho_l = 0.01$ for all l. In the Q-learning, the single-time-step

[3]Since the original reward setting in [2] is designed to be convenient for another selection, it has been altered to be suitable for softmax selection.

observation of $(y(t), \dot{y}(t), a(t), r(t+1), y(t+1), \dot{y}(t+1)) = (y, \dot{y}, a_j, r_k, y', \dot{y}')$ updated the l-th weight for action a_j by

$$w_{jl}^{(t+1)} = w_{jl}^{(t)} + \alpha \left(r_k + \gamma \max_{j' \in \mathcal{J}} Q_{j'}^{(t)}(y', \dot{y}') - Q_j^{(t)}(y, \dot{y}) \right) \phi_l(y, \dot{y}), \tag{31}$$

where $\mathcal{J} = \{1, 2, 3\}$ in this environment, denoting the three possible actions. For each trial, all the weights were initialized as $w_{jl}^{(1)} = 0$ for all j, l. Then, instead of (25), the derivative of the SC is approximated by

$$\zeta(\bar{t}) \triangleq -\sum_{l=1}^{L} \frac{\hat{v}_l^{(\bar{t})} \beta^{(\bar{t})}}{2 Z_l \left(\beta^{(\bar{t})}\right)^2} \times \sum_{j \in \mathcal{J}} \sum_{j' \in \mathcal{J}} \left(Q_j^{(\bar{t})}(\mu_l, \kappa_l) - Q_{j'}^{(\bar{t})}(\mu_l, \kappa_l) \right)^2 \times \exp\left(\beta^{(\bar{t})} \left(Q_j^{(\bar{t})}(\mu_l, \kappa_l) + Q_{j'}^{(\bar{t})}(\mu_l, \kappa_l) \right) \right), \tag{32}$$

where for all l,

$$\hat{v}_l^{(\bar{t})} \triangleq \frac{\sum_{j' \in \mathcal{J}} \left| w_{j'l}^{(t)} \right|}{\sum_{l=1}^{L} \sum_{j' \in \mathcal{J}} \left| w_{j'l}^{(t)} \right|}, \tag{33}$$

$$Z_l\left(\beta^{(\bar{t})}\right) \triangleq \sum_{j' \in \mathcal{J}} \exp\left(Q_{j'}^{(\bar{t})}(\mu_l, \kappa_l) \right). \tag{34}$$

We sampled 101 constant parameters at regular intervals from $[0, 2.5]$, which was determined to include the peak bandwidths of the conventional and proposed variable settings. For each of the sampled parameters, we ran 1000 trials using each variable setting and the Q-learning algorithm with $\gamma = 0.97$. Policy efficiency was measured as the total number of time steps per trial, averaged over 1000 trials. Note that the smaller the total number of time steps is, the higher is the policy efficiency.

B. Results

Figures8(a)–13(a) show the total number of time steps or the total reward per trial for the 20 best parameters (top 20 percent) over the sampling range. The 20 best parameters were ranked on the x-axis according to their results. Figures 8(b)–13(b) show the total number of time steps or the total reward per trial for each sampled parameter. The solid and broken lines in these figures were drawn, respectively, using our proposed setting and the conventional setting. The width of an error bar denotes the standard deviation of the results for each sampled parameter. In addition, for reference purposes only, we compared these settings with those for ϵ-greedy selection [2] with its timestep-invariant and timestep-variant parameters ϵ. Table I gives the total number of steps/the total reward per trial for the ϵ-greedy selection. The optimally tuned ϵ in the timestep-invariant ϵ-greedy selection was selected from 101 constant parameters sampled at regular intervals over $[0, 1]$, while the timestep-variant ϵ_t decayed according to $1 - h \times m(t)$, where h denotes the reciprocal of the number of episodes that was allowed in each trial; e.g., $1/100$ in the shortcut environment. Again, $m(t)$ denotes the number of times that the agent has reached a goal state, up to and including time step t.

TABLE I
TOTAL NUMBER OF TIME STEPS OR TOTAL REWARD PER TRIAL FOR ϵ-GREEDY SELECTION

environment	total number of time steps or total reward per trial	
	with timestep-invariant ϵ (optimally tuned ϵ)	with ϵ_t decayed according to $m(t)$
shortcut	920 ($\epsilon = 0.27$)	935
misleading	136 ($\epsilon = 0.8$)	140
deterministic	1045 ($\epsilon = 0.2$)	1051
entwined	427 ($\epsilon = 0.23$)	421
tic-tac-toe	50919 ($\epsilon = 0.07$)	39526
mountain–car	114929 ($\epsilon = 0.24$)	104536

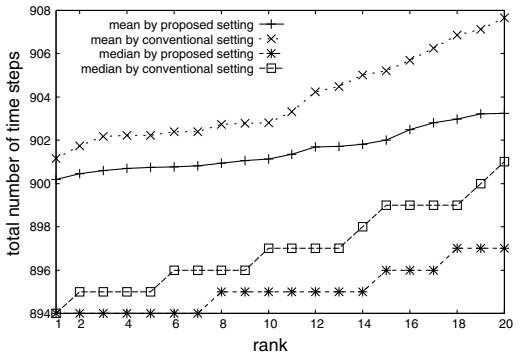
The peak bandwidth can be measured according to the total number of time steps or the total reward per trial for the 20 best parameters. From Figs. 8(a)–13(a), we see that our proposed setting gives a wider peak bandwidth than the conventional setting and is more effective, even with the best parameter for each setting. This is because the total number of time steps/total reward in our setting increases/decreases more gently and the optimal total number of time steps/total reward in our setting is small/large. In particular, Fig. 13 shows that our setting is suitable for extending the peak bandwidth even with a linear-function approximation of the action-value function estimates. Thus, our setting is less sensitive around the optimal parameter, which reduces the effort in tuning the parameter. The total number of time steps or total reward per trial over the sampling range, drawn in Figs. 8(b)–13(b), provides information on the peak performance for both settings. As expected, when defining our setting form in (23), the optimal parameters in both settings are mostly just greater than c. Hence, in our setting, we can reuse certain knowledge on tuning c from the conventional setting. Compared with the results in Table I for ϵ-greedy selection, softmax selection worked well in these environments regardless of the setting. We can see from the table that softmax selection generally yields adequate policies if it is tuned well. However, this does not mean that softmax selection always beats ϵ-greedy selection.

Next, using the error bars, we examined the policy stability of both settings. Particularly in the shortcut, deterministic, and tic-tac-toe environments, our setting yields more stable policies. Since the accuracy of tuning the parameter relies on policy stability, our setting is easier to handle.

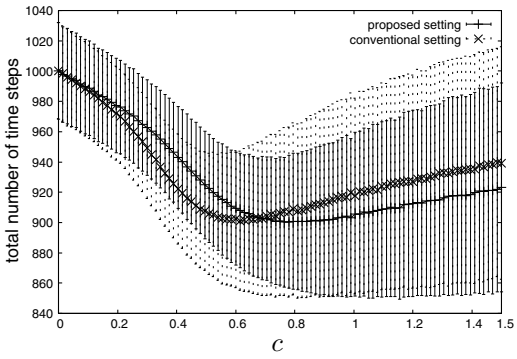
C. Statistical Evaluation through Multiple Comparisons

Figures 8–13 confirm that our proposed setting extends the peak bandwidth of the sampling range. To verify this statistically, we assessed the difference between the proposed and conventional methods in terms of policy efficiency around their respective peak bandwidths. The assessment was carried out in three steps.

1) Parametric versus nonparametric: Statistical tests are classified into two types, namely parametric and nonparametric

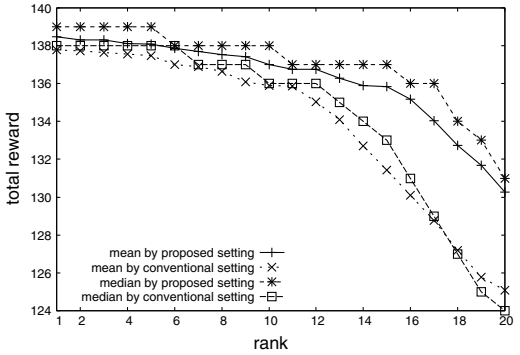


(a) Total number of time steps per trial for the 20 best parameters.

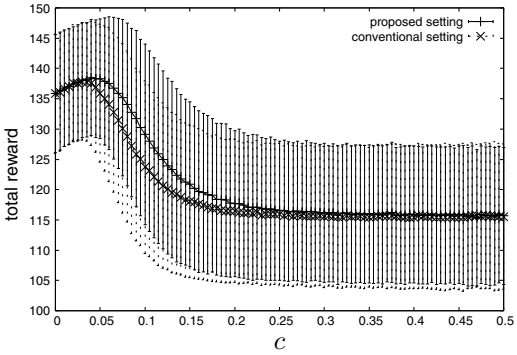


(b) Total number of time steps per trial over the sampling range.

Fig. 8. Results for the shortcut environment with the solid and broken lines depicting the proposed and conventional settings, respectively. The smaller the number of time steps is, the higher is the policy efficiency.



(a) Total reward per trial for the 20 best parameters.



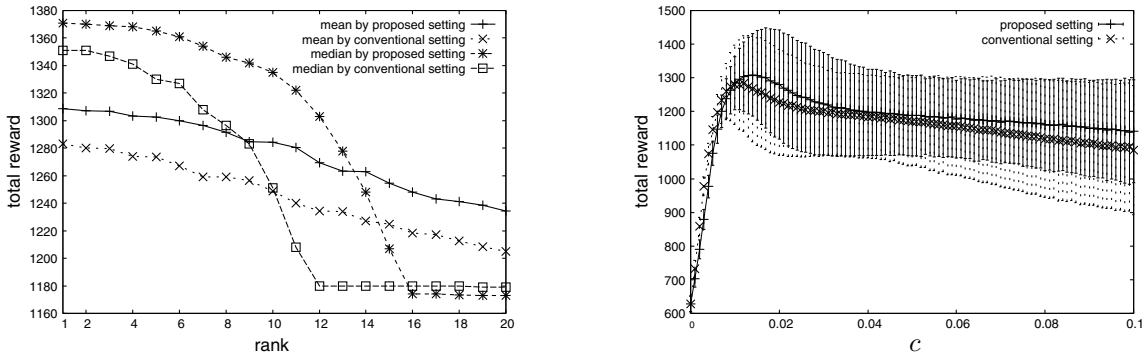
(b) Total reward per trial over the sampling range.

Fig. 9. Results for the misleading environment with the solid and broken lines depicting the proposed and conventional settings, respectively. The larger the total reward is, the higher is the policy efficiency.

tests [26], [27]. Roughly speaking, the former relies on the assumption that data points are normally distributed, whereas the latter does not. First, we needed to ascertain the normality of the distribution of data points at each c to determine the appropriate statistical test. In this case, a data point corresponds to the total number of time steps or total reward for a value of c in the sampling range. For example, boxplots for the shortcut and deterministic environments are given in Figs. 14 and 15 showing the respective distributions of data points. Owing to space limitations, we have omitted the boxplots for the other environments. For ease of visualization, we thinned out the sample points in the boxplots. The box in each boxplot is determined by the first and third quartiles of data points, with a horizontal line at the median value. The whiskers extend from the box ends for a range equal to 1.5 times the interquartile range. Data points lying outside the interval of the whiskers are regarded as outliers, and these are shown as circles. This method for drawing a boxplot is based on that in Gnuplot [28]. In each of the boxplots, there are so many outliers that most of the medians are far from the center of the box. The same can be said of the boxplots in the other environments. This suggests that the distribution of data points is far from the normal distribution, and hence, a nonparametric test would be more appropriate for testing the distribution.

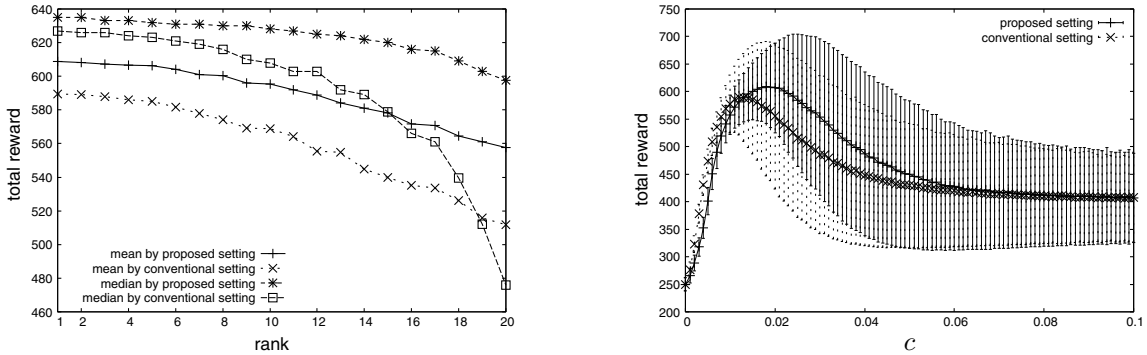
2) Kruskal–Wallis test: Next, we confirmed that the peak bandwidth exists in Figs. 8(b)–13(b). The Kruskal–Wallis test [26], [27] is an effective nonparametric test that uses multiple comparisons to ascertain whether there are any differences in the positions of distributions. In short, the positions of distributions denote the mean ranks of data points in the distributions. We applied the Kruskal–Wallis test under the null hypothesis that "the positions of distributions for all values of c in the sampling range are the same" and with a significance level of 0.01. The null hypothesis was rejected for all results shown in Figs. 8(b)–13(b), irrespective whether the conventional or proposed setting was used. This confirms that the peak bandwidth does indeed exist in each result.

3) Steel's test: Finally, we compared the conventional and proposed settings statistically in terms of their peak bandwidths. Steel's test [29] is one of the most useful multiple comparison nonparametric tests for this purpose. Here, the control in Steel's test corresponds to the set of data points for the best value of c (i.e., the best result), while the treatments correspond to the sets of data points for the other c values. Note that the number of treatments is 100 because there are 101 sampled points from the sampling range in each environment. Comparing the treatments with the control in a one-sided Steel's test yields a numerical evaluation of the peak



(a) Total reward per trial for the 20 best parameters. (b) Total reward per trial over the sampling range.

Fig. 10. Results for the deterministic environment with the solid and broken lines depicting the proposed and conventional settings, respectively. The larger the total reward is, the higher is the policy efficiency.



(a) Total reward per trial for the 20 best parameters. (b) Total reward per trial over the sampling range.

Fig. 11. Results for the entwined environment with the solid and broken lines depicting the proposed and conventional settings, respectively. The larger the total reward is, the higher is the policy efficiency.

bandwidth. Table II gives the number of treatments in each environment. Note that the number of treatments indicates how many treatments the test regarded as having no significant difference in the position of the distribution between these and the control. Clearly, the number of treatments measures the width of the peak bandwidth for c. In other words, the peak bandwidth for c is quantified through Steel's test with a significance level. The control used in obtaining the values for Table II is the value of c, for which the mean/median total number of time steps is minimal in the shortcut and mountain–car environments, and the mean/median total reward is maximal in the other environments. A significance level of 0.01 was applied in the test. From Table II, we see that our proposed setting yields a wider peak bandwidth from a perspective of statistical significance. Incidentally, it is more important to note which of the two numbers in the table is greater rather than how large the difference is between the two numbers. This is because the difference depends strongly on the significance level and the sampling density, whereas which of the two numbers is larger does not depend on these.

TABLE II
NUMBER OF TREATMENTS WITH THE OPTIMAL CONTROL IN TERMS OF THE MEAN/MEDIAN TOTAL NUMBER OF TIME STEPS OR TOTAL REWARD.

environment	conventional setting	proposed setting
	number of treatments in terms of mean/median	
shortcut	11 / 12	**20 / 17**
misleading	4 / **5**	**5 / 5**
deterministic	7 / 7	**9 / 9**
entwined	10 / 4	**12 / 7**
tic-tac-toe	24 / 23	**25 / 25**
mountain–car	23 / 23	**41 / 34**

V. DISCUSSION AND CONCLUSION

We focused on a linear form for variable $\beta^{(t)}$ in (11). Similar results were obtained using the exponential form,

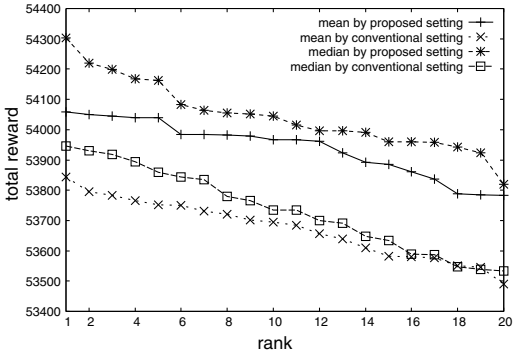
$$\beta^{(t)} = c^{m(t)} - 1, \tag{35}$$

where $c \geq 1$ is a constant parameter that needs to be tuned. Our setting for this form is expressed as

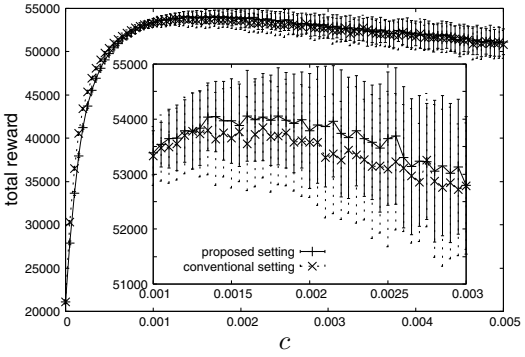
$$\beta^{(t)} = c^{m(t)+b(m(t))} - 1. \tag{36}$$

For example, Fig. 16 shows the results for the misleading environment, giving both the proposed and conventional settings. The same settings as those used for the linear form were employed with the exponential form. According to the figure, the proposed setting clearly succeeds in extending the peak bandwidth.

In this paper, we improved the variable setting of softmax selection by extending the peak bandwidth using the derivative

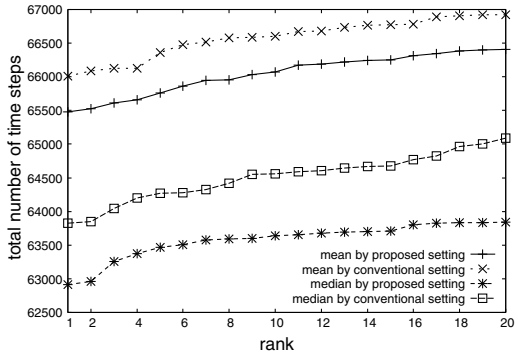


(a) Total reward per trial for the 20 best parameters.

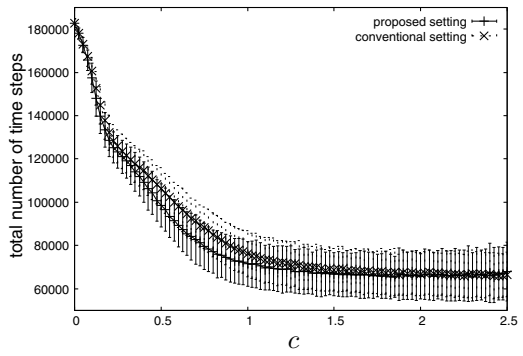


(b) Total reward per trial over the sampling range.

Fig. 12. Results for the tic-tac-toe environment with the solid and broken lines depicting the proposed and conventional settings, respectively. The larger the total reward is, the higher is the policy efficiency.



(a) Total number of time steps per trial for the 20 best parameters.



(b) Total number of time steps per trial over the sampling range.

Fig. 13. Results for the mountain–car environment with the solid and broken lines depicting the proposed and conventional settings, respectively. The smaller the number of time steps is, the higher is the policy efficiency.

of the SC. Our setting, designed to control the typical set of sequences, is both simple and easy to implement. Moreover, using a variety of episodic tasks, we confirmed that it is more effective in extending the peak bandwidth and yields a better policy in terms of stability.

Compared with timestep-by-timestep settings, the episode-by-episode settings discussed in this paper are better suited to the stationarity and ergodicity assumptions for the AEP, because the policy is not affected by changes in β, at least during each episode. This work thus focused on episodic tasks. Testing the timestep-by-timestep version of our setting remains as future work. Other future work involves testing our setting for continuous state-actions using various function approximation methods such as those discussed in [30], [31].

APPENDIX A
PROOF OF THE SC DERIVATIVE

Assume that v_i is invariant to β. This implies that v_i does not immediately change with β. Differentiating $H(P)$ in (14) with respect to β under this assumption gives

$$\frac{dH(P)}{d\beta} = \sum_{i=1}^{I} v_i \sum_{j=1}^{J} (-\log p_{ij} - 1) \frac{\partial p_{ij}}{\partial \beta}. \tag{37}$$

Using

$$p_{ij} = \frac{\exp(\beta Q_{ij})}{Z_i(\beta)} \quad \text{and} \quad Z_i(\beta) = \sum_{j=1}^{J} \exp(\beta Q_{ij}), \tag{38}$$

where $Q_{ij} = 0$ for $j \notin \mathcal{J}_i$, we obtain

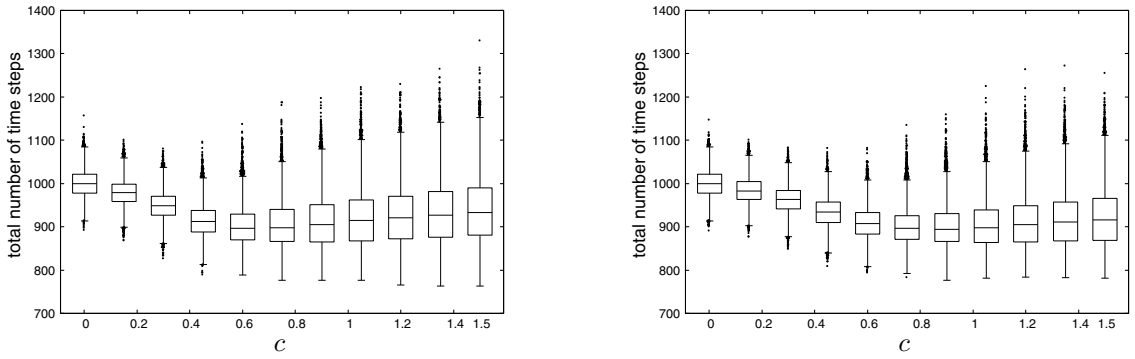
$$\frac{dH(P)}{d\beta} = \sum_{i=1}^{I} v_i \sum_{j=1}^{J} (-\beta Q_{ij} + \log Z_i(\beta) - 1) \times \frac{\left(Q_{ij} Z_i(\beta) \exp(\beta Q_{ij}) - \exp(\beta Q_{ij}) \frac{\partial Z_i(\beta)}{\partial \beta} \right)}{Z_i(\beta)^2}, \tag{39}$$

where $\partial Z_i(\beta)/\partial\beta$ is

$$\frac{\partial Z_i(\beta)}{\partial \beta} = \sum_{j=1}^{J} Q_{ij} \exp(\beta Q_{ij}). \tag{40}$$

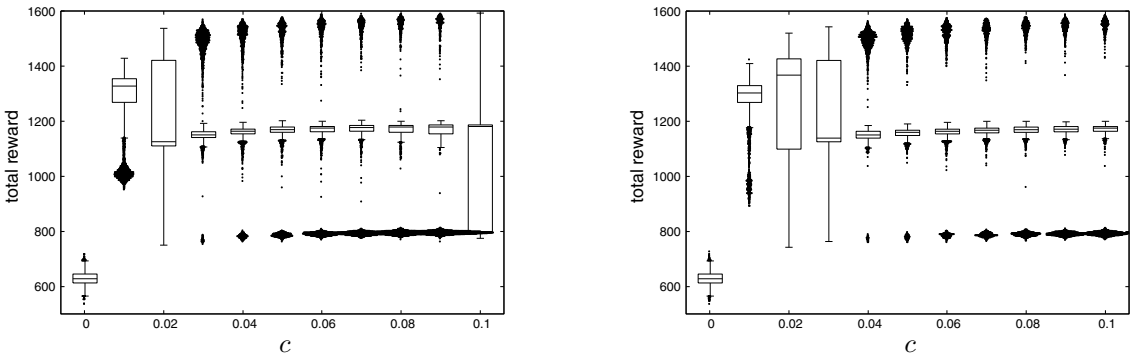
This equation can be rewritten as

$$\frac{dH(P)}{d\beta} = -\sum_{i=1}^{I} \frac{v_i \beta}{Z_i(\beta)^2} \times \left\{ Z_i(\beta) \sum_{j=1}^{J} Q_{ij}^2 \exp(\beta Q_{ij}) - \left(\frac{\partial Z_i(\beta)}{\partial \beta} \right)^2 \right\}. \tag{41}$$



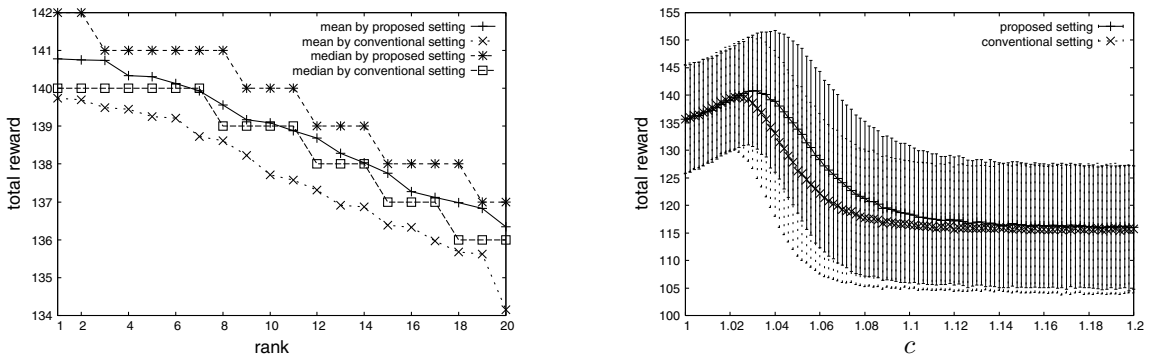
(a) Conventional setting. (b) Proposed setting.

Fig. 14. Boxplot for the shortcut environment illustrating the statistical distribution of the total number of time steps.



(a) Conventional setting. (b) Proposed setting.

Fig. 15. Boxplot for the deterministic environment illustrating the statistical distribution of total reward.



(a) Total reward per trial for the 20 best parameters. (b) Total reward per trial over the sampling range.

Fig. 16. Results for the misleading environment obtained using the exponential form, with the solid and broken lines depicting the proposed and conventional settings, respectively. The larger the total reward is, the higher is the policy efficiency.

Since

$$Z_i(\beta)\sum_{j=1}^{J} Q_{ij}^2 \exp\left(\beta Q_{ij}\right) = \frac{1}{2}\sum_{j=1}^{J}\exp\left(\beta Q_{ij}\right)\sum_{j'=1}^{J} Q_{ij'}^2 \exp\left(\beta Q_{ij'}\right) + \frac{1}{2}\sum_{j'=1}^{J}\exp\left(\beta Q_{ij'}\right)\sum_{j=1}^{J} Q_{ij}^2 \exp\left(\beta Q_{ij}\right), \quad (42)$$

and

$$\left(\frac{\partial Z_i(\beta)}{\partial \beta}\right)^2 = \sum_{j=1}^{J} Q_{ij}\exp\left(\beta Q_{ij}\right)\sum_{j'=1}^{J} Q_{ij'}\exp\left(\beta Q_{ij'}\right), \quad (43)$$

we reach the SC derivative in (22).

APPENDIX B
Q-LEARNING UPDATE IN THE TIC-TAC-TOE GAME

Suppose that an agent's turn results in the placement of marks as shown on the leftmost board in Fig. 17, where the

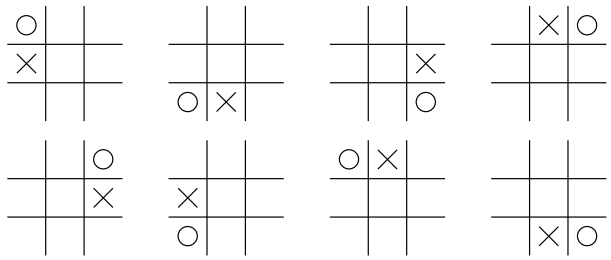


Fig. 17. Rotations of 0, 90, 180, and 270 degrees, and reflections with respect to the vertical, horizontal, and both diagonal axes.

circles and crosses denote the agent's and opponent's marks, respectively.

This placement is represented as a state-action. Additionally, suppose that the subsequent opponent's turn results in marking the center space, which yields a reward. This marking and reward process corresponds to a state-transition produced by the state-action. The Q-learning update with the state-transition is applied not only to the state-action but also to the state-actions of its rotations and reflections. As illustrated in Fig. 17, there are eight state-actions in total, namely the state-action itself together with the rotations and reflections thereof.

ACKNOWLEDGMENT

We wish to thank Mr. Kenji Ono, Mr. Shun Igo, Prof. Nobuo Suematsu, and Prof. Akira Hayashi for their insightful comments on the preliminary work [32] of this paper.

REFERENCES

[1] T. M. Mitchell, *Machine Learning*, ser. McGraw-Hill computer science series. New York: McGraw-Hill, 1997.
[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, ser. Adaptive Computation and Machine Learning. Cambridge, MA: MIT Press, Mar. 1998.
[3] M. A. Walker, "An application of reinforcement learning to dialogue strategy selection in a spoken dialogue system for email," *Journal of Artificial Intelligence Research*, vol. 12, pp. 387–416, Jun. 2000.
[4] Y.-C. Wang and J. M. Usher, "Application of reinforcement learning for agent-based production scheduling," *Engineering Applications of Artificial Intelligence*, vol. 18, no. 1, pp. 73–82, Feb. 2005.
[5] J. Morimoto and K. Doya, "Robust reinforcement learning," *Neural Computation*, vol. 17, no. 2, pp. 335–359, Feb. 2005.
[6] P. Varshavskaya, L. P. Kaelbling, and D. Rus, "Automated design of adaptive controllers for modular robots using reinforcement learning," *The International Journal of Robotics Research*, vol. 27, no. 3–4, pp. 505–526, Mar. 2008.
[7] T. Mahmood and F. Ricci, "Improving recommender systems with adaptive conversational strategies," in *Proceedings of the 20th ACM conference on Hypertext and hypermedia*. New York, NY: ACM, Jun. 2009, pp. 73–82.
[8] I. Grondman, L. Buşoniu, G. A. D. Lopes, and R. Babuška, "A survey of actor-critic reinforcement learning: Standard and natural policy gradients," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 42, no. 6, pp. 1291–1307, Nov. 2012.
[9] K.-S. Hwang and C.-Y. Lo, "Policy improvement by a model-free Dyna architecture," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 3, pp. 776–788, May 2013.
[10] C. J. C. H. Watkins and P. Dayan, "Technical note: Q-learning," *Machine Learning*, vol. 8, pp. 279–292, 1992.
[11] P. Dayan, "The convergence of TD(λ) for general λ," *Machine Learning*, vol. 8, no. 3, pp. 341–362, May 1992.
[12] S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvári, "Convergence results for single-step on-policy reinforcement learning algorithms," *Machine Learning*, vol. 38, no. 3, pp. 287–308, Mar. 2000.
[13] R. Dearden, N. Friedman, and S. Russell, "Bayesian Q-learning," in *Proceedings of the 15th National Conference on Artificial Intelligence*, American Association for Artificial Intelligence. Madison, Wisconsin: AAAI Press, Jul. 1998, pp. 761–768.
[14] S. Ishii, W. Yoshida, and J. Yoshimoto, "Control of exploitation-exploration meta-parameter in reinforcement learning," *Neural Networks*, vol. 15, no. 4–6, pp. 665–687, June–July 2002.
[15] K. Iwata, K. Ikeda, and H. Sakai, "A new criterion using information gain for action selection strategy in reinforcement learning," *IEEE Transactions on Neural Networks*, vol. 15, no. 4, pp. 792–799, Jul. 2004.
[16] C. Chen, D. Dong, H.-X. Li, J. Chu, and T.-J. Tarn, "Fidelity-based probabilistic Q-learning for control of quantum systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 5, pp. 920–933, May 2014.
[17] R. Lincoln, S. Galloway, B. Stephen, and G. Burt, "Comparing policy gradient and value function based reinforcement learning methods in simulated electrical power trade," *IEEE Transactions on Power Systems*, vol. 27, no. 1, pp. 373–380, Feb. 2012.
[18] K. Iwata, K. Ikeda, and H. Sakai, "The asymptotic equipartition property in reinforcement learning and its relation to return maximization," *Neural Networks*, vol. 19, no. 1, pp. 62–75, Jan. 2006.
[19] K. Iwata, "An information-theoretic analysis of return maximization in reinforcement learning," *Neural Networks*, vol. 24, no. 10, pp. 1074–1081, Dec. 2011.
[20] R. S. Sutton, "Temporal credit assignment in reinforcement learning," PhD thesis, University of Massachusetts, Amherst, MA, 1984.
[21] L. P. Kaelbling, *Learning in Embedded Systems*. Cambridge, MA: MIT Press, 1993.
[22] H. J. Kushner and G. G. Yin, *Stochastic Approximation Algorithms and Applications*, ser. Applications of Mathematics. New York: Springer-Verlag, 1997, vol. 35.
[23] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. New York: John Wiley & Sons, Inc., 2006.
[24] C. J. C. H. Watkins, "Learning from delayed rewards," PhD thesis, King's College, Cambridge, UK, 1989.
[25] M. Geist and O. Pietquin, "Algorithmic survey of parametric value function approximation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 6, pp. 845–867, Jun. 2013.
[26] J. D. Gibbons and S. Chakraborti, *Nonparametric Statistical Inference*, 5th ed. Boca Raton, FL: CRC Press, 2011.
[27] A. Field, J. Miles, and Z. Field, *Discovering Statistics Using R*. London, UK: Sage, 2012.
[28] T. Williams and C. Kelley, *gnuplot 4.6: An Interactive Plotting Program*, 2014. [Online]. Available: http://sourceforge.net/projects/gnuplot/
[29] R. G. D. Steel, "A multiple comparison rank sum test: Treatments versus control," *Biometrics*, vol. 15, no. 4, pp. 560–572, Dec. 1959.
[30] X. Xu, Z. Huang, D. Graves, and W. Pedrycz, "A clustering-based graph laplacian framework for value function approximation in reinforcement learning," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2613–2625, Dec. 2014.
[31] D. Zhao and Y. Zhu, "MEC – a near-optimal online reinforcement learning algorithm for continuous deterministic systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 2, pp. 346–356, Feb. 2015.
[32] K. Ono, K. Iwata, and A. Hayashi, "An action-selection strategy insensitive to parameter-settings in reinforcement learning," in *Proceedings of ICROS-SICE International Joint Conference 2009*, SICE and ICROS. Fukuoka, Japan: SICE, Aug. 2009, pp. 1012–1017.

Kazunori Iwata Dr Kazunori Iwata received a BE and an ME degree from Nagoya Institute of Technology, Aichi, Japan in 2000 and 2002, respectively, and a PhD degree in Informatics from Kyoto University, Kyoto, Japan, in 2005. He was a JSPS research fellow from April 2002 to March 2005. He has been with Hiroshima City University, Hiroshima, Japan, since April 2005. His research interests include machine learning, pattern recognition, and information theory. He currently serves as an Associate Editor for *IEICE Transactions on Information and Systems*. He is a member of the IEEE and IEICE.