PAPER

Reinforcement Learning with Orthonormal Basis Adaptation Based on Activity-Oriented Index Allocation

Hideki SATOH[†], *Member*

SUMMARY An orthonormal basis adaptation method for function approximation was developed and applied to reinforcement learning with multi-dimensional continuous state space. First, a basis used for linear function approximation of a control function is set to an orthonormal basis. Next, basis elements with small activities are replaced with other candidate elements as learning progresses. As this replacement is repeated, the number of basis elements with large activities increases. Example chaos control problems for multiple logistic maps were solved, demonstrating that the method for adapting an orthonormal basis can modify a basis while holding the orthonormality in accordance with changes in the environment to improve the performance of reinforcement learning and to eliminate the adverse effects of redundant noisy states.

key words: *orthonormal basis, function approximation, nonlinear, reinforcement learning, activity*

1. Introduction

Various methods have been developed for function approximation and reinforcement learning with a multi-dimensional continuous state space [1]. The wire fitting approach performs function approximation using control wires, which fit the control surface and create the relationship between the state and the action [2]. Tile coding allocates tiles whose position and size are randomly selected in a state space and use the tiles as a basis [1]. A tree-based algorithm has been developed to handle a large continuous state space [3]. These methods work well for an environment with a multi-dimensional continuous state space. However, performing reinforcement learning becomes more difficult as the dimension of a state space increases because the number of elements in a basis for function approximation geometrically increases with respect to the dimension of the state space. This problem, which is the so-called "curse of dimensionality", is one of the most serious problems of function approximation and reinforcement learning with a high-dimensional state space [1].

Methods using a radial basis function (RBF) [1] as an element of a basis have attracted much interest for solving the curse of dimensionality because the basis that is based on RBF can be continuously modified by adjusting the parameters of RBF, and thus RBF is suitable for adaptively constructing a basis as progresses learning. One approach to modifying the basis is an adaptive basis division algorithm, which divides the RBF according to the statistical properties of the temporal difference (TD) error [4]. Another one is a method that adjusts the mean and covariance of RBF using gradient based adaptation and cross entropy based adaptation [5]. Laplacian approaches have been developed, which connect nearby samples with respect to the state space geometry using the graph Laplacian, and basis elements are set to the eigen vectors of the Laplacian [6], [7]. These methods have to perform eigen analysis, and thus their calculation costs are very high. A basis adaptation method based on an evolutionary state recruitment strategy was developed to reduce the calculation costs [8].

Another serious problem in reinforcement learning is that a large number of trials need to be repeated to achieve progress in learning. Thus, a long time is necessary to finish the learning, or equipment to be controlled may not endure the trials if it is composed of mechanical parts, like a robot is. Hybrid learning solves this problem [9], [10]. An approach to hybrid learning first makes a control function using a non-linear control theory [11], [12], next the control function is approximated using linear function approximation, and the approximated control function is finally improved using reinforcement learning. Although obtaining a control function that works well for an arbitrary non-linear environment is very difficult, the control function obtained in the first step is allowed to be a rough approximation, and even a control function obtained by a linear control theory [13] and a linear approximation of an environment may be useful. Thus, hybrid learning is very effective to reduce the number of trials if an explicit equation of an environment is known in advance.

If basis adaptation can be used with hybrid learning, the possibility of solving more complicated tasks in actual systems should be increased. However, applying basis adaptation to hybrid learning has some requirements for actual systems. The first requirement is that the basis that works for function approximation in the second step of hybrid learning must be given in advance. The second one is that the basis should be an orthogonal one. This is because we can approximate the control function in the second step by Fourier series expansion using an orthogonal basis more easily and with less approximation error compared with a case in which

Manuscript received August 1, 2007.
Manuscript revised November 22, 2007.

[†]The author is with the Future University-Hakodate, Hakodate-shi, 041-8655 Japan.

a non-orthogonal basis such as RBF is used. The third one is robustness. That is, we need not only to extract the effective dynamics in the states but also to eliminate the adverse effect of unknown noise contained in the states. This is because actual systems with a large number of states often contain unknown noise or states that are independent of the performance.

Unfortunately, conventional methods do not satisfy all the requirements. A method for adapting an orthonormal basis that is based on activity-oriented index allocation was thus developed and applied to reinforcement learning. The method is presented in this paper, and the method is shown to satisfy these three requirements and to reduce the number of basis elements required to perform reinforcement learning even if the environment changes and contains disturbance noise.

2. Reinforcement Learning in Continuous State Space

A system with reinforcement learning [1] is divided into two parts: agents and an environment. The former provide control functions, and the latter is the target system to be controlled. An agent observes the state and the reward from the environment, updates a control function accordingly, and outputs a new control input to the environment. The control function and control input are referred to as policy and action, respectively. Actor-critic methods [1] were developed to perform reinforcement learning, and they are very interesting because of two advantages: They require minimum computation to select actions, so the implementation of the controller is very easy. Moreover, they can learn an explicit stochastic policy, so they are useful in competitive and non-Markov cases [1]. Therefore, actor-critic methods were selected from various machine learning methods. An implementation of the actor-critic method is summarized in this section [14].

Consider a discrete-time continuous-state continuous-action environment. Let s_d be the dth element of the state, $d_{\rm s}$ be the dimension of the state, $\boldsymbol{s} \overset{\rm def}{=} (s_1, \cdots, s_{d_{\rm s}})^{\rm t}$, u_d be the dth element of the action, $d_{\rm u}$ be the dimension of the action, $\boldsymbol{u} \overset{\rm def}{=} (u_1, \cdots, u_{d_{\rm u}})^{\rm t}$, and superscript t denote transposition. The actor-critic method consists of two parts: an actor and a critic. The actor observes $\boldsymbol{s}_t$, the state at time t, and decides on $\boldsymbol{u}_t$, the action at time t, which is a sample value from the conditional Gaussian distribution with density $p(\boldsymbol{u}_t|\boldsymbol{s}_t)$ defined by

$$p(\boldsymbol{u}|\boldsymbol{s}) \overset{\rm def}{=} \prod_{d=1}^{d_{\rm u}} p_d(u_d|\boldsymbol{s}), \tag{1}$$

$$p_d(u_d|\boldsymbol{s}) \overset{\rm def}{=} \frac{1}{\sqrt{2\pi}\sigma_d(\boldsymbol{s})} \exp(\frac{-(u_d-\mu_d(\boldsymbol{s}))^2}{2\sigma_d(\boldsymbol{s})^2}), \tag{2}$$

where $p_d(u_d|\boldsymbol{s})$ is a one-dimensional conditional Gaussian density function with mean $\mu_d(\boldsymbol{s})$ and standard deviation $\sigma_d(\boldsymbol{s})$.

The critic observes reward r_t and updates $\mu_d(\boldsymbol{s})$ and $\sigma_d(\boldsymbol{s})$ for $1 \le d \le d_{\rm u}$ so that discount return R_t, defined by

$$R_t \overset{\rm def}{=} \sum_{t'=0}^{\infty} {\nu_{\rm DR}}^{t'} r_{t+t'+1},$$

is maximized, where $\nu_{\rm DR}$ is the discount rate ($0 \le \nu_{\rm DR} \le 1$). For $1 \le d \le d_{\rm u}$, $\mu_d(\boldsymbol{s})$ and $\sigma_d(\boldsymbol{s})$ are expressed using the following function approximations:

$$\mu_d(\boldsymbol{s}) \overset{\rm def}{=} \sum_{i=0}^{N} \xi_{di}\phi_i(\boldsymbol{s}), \tag{3}$$

$$\sigma_d(\boldsymbol{s}) \overset{\rm def}{=} h_{\rm limit}(\sum_{i=0}^{N}(\eta_{di}\phi_i(\boldsymbol{s})), \tag{4}$$

where $\{\phi_i(\boldsymbol{s})\}$ is a basis and $h_{\rm limit}(\cdot)$ is a function to limit $\sigma_d(\boldsymbol{s})$ to a positive value. In this paper, $h_{\rm limit}(\cdot)$ is given by

$$h_{\rm limit}(x) \overset{\rm def}{=} \frac{\sigma_{\rm umax} - \sigma_{\rm umin}}{1+\exp(-x)} + \sigma_{\rm umin}, \tag{5}$$

where $\sigma_{\rm umin}$ and $\sigma_{\rm umax}$ denote the upper and lower bounds of $\sigma_d(\boldsymbol{s})$. Equation (3) with a sufficiently large N ($> d_{\rm s}$) can approximate the nonlinear relationship in the control function to a linear relationship between $(\phi_0(\boldsymbol{s}), \cdots, \phi_N(\boldsymbol{s}))^{\rm t}$ and $\mu_d(\boldsymbol{s})$. Vector $(\phi_0(\boldsymbol{s}), \cdots, \phi_N(\boldsymbol{s}))^{\rm t}$ is referred to as a feature vector.

The parameters of $\mu_d(\boldsymbol{s})$ and $\sigma_d(\boldsymbol{s})$ ($\xi_{d0}, \cdots, \xi_{dN}$ and $\eta_{d0}, \cdots, \eta_{dN}$) are initially set so that $\boldsymbol{u}_t$ is distributed in the whole definition domain. Thus, the critic can learn the relationships between r_t, $\boldsymbol{s}_t$, and $\boldsymbol{u}_t$. As the critic progresses in the learning, $\mu_d(\boldsymbol{s})$ and $\sigma_d(\boldsymbol{s})$ are updated so that R_t increases. The update of $\mu_d(\boldsymbol{s})$ and $\sigma_d(\boldsymbol{s})$ can be done by adjusting parameters $\xi_{d0}, \cdots, \xi_{dN}$ and $\eta_{d0}, \cdots, \eta_{dN}$ using temporal-difference (TD) learning [1] and the steepest descent method.

When the dimension of $\boldsymbol{s}_t$ is high, the degree of expansion of $\mu_d(\boldsymbol{s})$ and $\sigma_d(\boldsymbol{s})$, N, is too large to perform reinforcement learning. This curse of dimensionality is one of the most serious problems of function approximation and reinforcement learning with a high-dimensional state space.

3. Adapting an Orthonormal Basis Based on Activity-Oriented Index Allocation

If the degree of expansion of $\mu_d(\boldsymbol{s})$ and $\sigma_d(\boldsymbol{s})$, N, is too large, we cannot practically perform reinforcement learning because of the calculation cost. However, we may not obtain the required accuracy if N is small. Thus, basis $\{\phi_i(\boldsymbol{s})\}$ in Eqs. (3) and (4) should be selected carefully to obtain the required accuracy by using a limited value of N. The activity-oriented index allocation method (AIA) is presented in this paper; it can adaptively modify an orthonormal basis in accordance with the changes in the environment so that reinforcement learning works well even if N is small.

Let $\{K(\boldsymbol{s},\boldsymbol{k})\}$ be a multi-dimensional orthonormal basis with its elements defined by

$$K(\boldsymbol{s},\boldsymbol{k}) \overset{\text{def}}{=} \prod_{d=1}^{d_{\text{s}}} K_d(s_d,k_d), \tag{6}$$

$$K_d(s_d,k_d) \overset{\text{def}}{=} \begin{cases} \sqrt{\dfrac{1}{\Delta_{\text{s}d}}} \text{ for } k_d=0 \\ \sqrt{\dfrac{2}{\Delta_{\text{s}d}}} \sin(\dfrac{k_d+1}{2}\omega_{0d}(s_d-s_{\min d})) \text{ for } k_d=1,3,\cdots, \\ \sqrt{\dfrac{2}{\Delta_{\text{s}d}}} \cos(\dfrac{k_d}{2}\omega_{0d}(s_d-s_{\min d})) \text{ for } k_d=2,4,\cdots \end{cases}$$

where $\omega_{0d} \overset{\text{def}}{=} 2\pi/\Delta_{\text{s}d}$, $\Delta_{\text{s}d} \overset{\text{def}}{=} s_{\max d}-s_{\min d}$, $s_{\max d}$ is the maximum value of s_d, $s_{\min d}$ is the minimum value of s_d, $\boldsymbol{k} \overset{\text{def}}{=} (k_1,\cdots,k_{d_{\text{s}}})^{\text{t}}$, and $\boldsymbol{k}$ is referred to as the index vector. Let $\phi_i(\boldsymbol{s})$ be defined as

$$\phi_i(\boldsymbol{s}) \overset{\text{def}}{=} K(\boldsymbol{s},\boldsymbol{k}), \tag{7}$$

where i is referred to as the index of the basis. Let $\mathcal{D}_{\boldsymbol{k}}$ denote a set of $\boldsymbol{k}$. When $\mathcal{D}_{\text{k}d} \overset{\text{def}}{=} \{0,1,\cdots,N_d\}$ and $\mathcal{D}_{\boldsymbol{k}}$ is given by the Cartesian product as $\mathcal{D}_{\boldsymbol{k}} = \mathcal{D}_{\text{k}1} \times \mathcal{D}_{\text{k}2}\times,\cdots,\times\mathcal{D}_{\text{k}d_{\text{s}}}$, the relationship between $\boldsymbol{k}$ and i can be obtained using

$$i=\sum_{d=1}^{d_{\text{s}}} k_d \prod_{d'=d+1}^{d_{\text{s}}} N_{d'}, \tag{8}$$

and N is given by

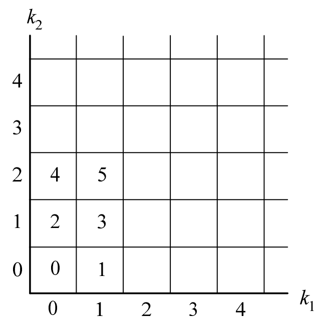
$$N=\prod_{d=1}^{d_{\text{s}}}(N_d+1)-1. \tag{9}$$

Equation (9) cleary indicates that N geometrically increases with respect to d_{s}. Thus, if $\mathcal{D}_{\boldsymbol{k}}$ is given by the Cartesian product of $\mathcal{D}_{\text{k}d}$ and if d_{s} is large, N is too large to perform reinforcement learning. This is the curse of dimensionality.

The method for adapting an orthonormal basis that is based on AIA presented in this section is a solution for the curse of dimensionality. Let us introduce an index table, *IDXT*, defined by an $(N+1)\times d_{\text{s}}$ matrix to express the relationship between i and $\boldsymbol{k}$. An example *IDXT*, when $\mathcal{D}_{\boldsymbol{k}}$ is given by the Cartesian product, $d_{\text{s}}=2$, $N_1=1$, and $N_2=2$, is shown in Fig. 1-a. Consider k_1, $\cdots$, $k_{d_{\text{s}}}$ to be the coordinates with respect to a rectangular coordinate system, which is referred to as the $\boldsymbol{k}$-coordinate system. Index i in Fig. 1-a is expressed in the $\boldsymbol{k}$-coordinate system, as shown in Fig. 1-b.

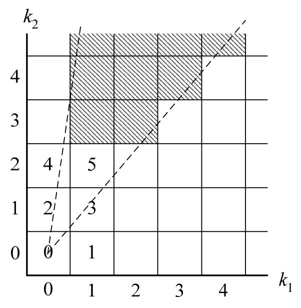
i \ k_d	k_1	k_2
0	0	0
1	1	0
2	0	1
3	1	1
4	0	2
5	1	2

(a) *IDXT*

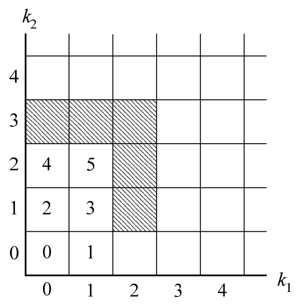


(b) Index i in $\boldsymbol{k}$-coordinate system

Fig. 1 Index table *IDXT* and index i in $\boldsymbol{k}$-coordinate system.



(a) DA-method



(b) NA-method

Fig. 2 Examples of search space of DA- and NA-methods.

Although the accuracies of Eqs. (3) and (4) increase as N increases, we cannot use an arbitrarily large value of N. Therefore, when N and $\{K(\boldsymbol{s},\boldsymbol{k})\}$ are given, the elements in $\{K(\boldsymbol{s},\boldsymbol{k})\}$ that affect the accuracy need to be identified. On the basis of this viewpoint, the activity of an element in $\{K(\boldsymbol{s},\boldsymbol{k})\}$ is proportional to $\|\boldsymbol{\xi}\|_i$ defined by

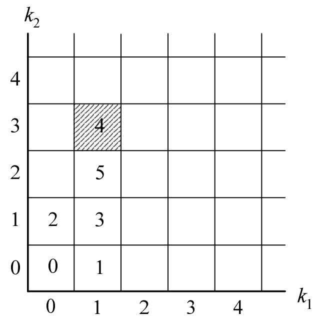
$$\|\boldsymbol{\xi}\|_i \overset{\text{def}}{=} \sum_{d=1}^{d_{\text{u}}} \xi_{di}^2, \tag{10}$$

and AIA updates *IDXT* so that $\|\boldsymbol{\xi}\|_i$ for $0\le i\le N$ is as large as possible. Let $\|\boldsymbol{\xi}\|_{i_{\text{small}}}$ be the smallest one, $\|\boldsymbol{\xi}\|_{i_{\text{large}}}$ be the largest one, the index vector corresponding to i_{small} be $\boldsymbol{k}_{\text{small}}$, and the index vector corresponding to i_{large} be $\boldsymbol{k}_{\text{large}}$. The elements in the i_{small}th row of *IDXT*, $\boldsymbol{k}_{\text{small}}$, are replaced with those in $\boldsymbol{k}_{\text{new}}$.

An implementation of AIA sets $\boldsymbol{k}_{\text{new}}$ to the index vector closest to origin $\mathbf{0}$ in the search space that is a conical space spreading in the $\boldsymbol{k}_{\text{large}}$ direction in the $\boldsymbol{k}$-coordinate system. This is referred to as the direction-based allocation method (DA-method). Another implementation of AIA sets $\boldsymbol{k}_{\text{new}}$ to an index vector randomly selected in the search space that is a square space around $\boldsymbol{k}_{\text{large}}$ in the $\boldsymbol{k}$-coordinate system. This is referred to as the neighborhood allocation method (NA-method). Note that the index vectors that are already in *IDXT* are eliminated from the search spaces. Examples of the search spaces when $i_{\text{large}}=5$ are shown by the shaded areas in Fig. 2. An example of *IDXT* after

i \ k_d	k_1	k_2
0	0	0
1	1	0
2	0	1
3	1	1
4	▨ 1	▨ 3
5	1	2

(a) *IDXT*



(b) Index i in $\boldsymbol{k}$-coordinate system

Fig. 3 Index table *IDXT* and index i in $\boldsymbol{k}$-coordinate system after updating.

updating when $i_{\text{small}} = 4$ and $i_{\text{large}} = 5$ is shown in Fig. 3-a. The shaded area in Fig. 3-a is the updated index vector, $\boldsymbol{k}_{\text{new}}$. Index i_{small} in the $\boldsymbol{k}$-coordinate system after updating is shown by the shaded area in Fig. 3-b.

The DA- and NA-methods described above work well for the following reasons:

(1) Consider the case of $d_{\text{s}} = 2$. Let us rewrite ξ_{di} as $\xi_{d\boldsymbol{k}}$ by replacing i with $\boldsymbol{k} = (k_1, k_2)^{\text{t}}$ and rewrite Eq. (3) as

$$\mu_d(\boldsymbol{s}) = \sum_{k_1,k_2} \xi_{d\boldsymbol{k}} K(\boldsymbol{s}, \boldsymbol{k}) \tag{11}$$

by replacing $\phi_i(\boldsymbol{s})$ with $K(\boldsymbol{s}, \boldsymbol{k})$. If $k_{\text{large1}} > 0$ and $k_{\text{large2}} > 0$, both s_1 and s_2 affect $\mu_d(\boldsymbol{s})$, and the principal relationship between them is expressed by $K(\boldsymbol{s}, \boldsymbol{k}_{\text{large}})$. The relationship is more precisely expressed using $K(\boldsymbol{s}, \boldsymbol{k}_{\text{large}})$ and $K(\boldsymbol{s}, \boldsymbol{k}')$ such that $\boldsymbol{k}'$ is close to $\boldsymbol{k}_{\text{large}}$ compared with a case in which $\boldsymbol{k}'$ is far from $\boldsymbol{k}_{\text{large}}$. This is because the probability that $K(\boldsymbol{s}, \boldsymbol{k}')$ is independent of the relationship expressed by $K(\boldsymbol{s}, \boldsymbol{k}_{\text{large}})$ increases as the distance between $\boldsymbol{k}'$ and $\boldsymbol{k}_{\text{large}}$ increases.

(2) This point can be understood in a more concrete manner by considering a case where s_2 is independent of increasing the reward. In this case, $K(\boldsymbol{s}, \boldsymbol{k})$ with $k_2 > 0$ does not contribute to increasing the reward, so the critic updates $\xi_{d(k_1,0)^{\text{t}}}$ while $\xi_{d(k_1,k_2)^{\text{t}}}$ for $k_2 > 0$ remains zero. Therefore, setting $\{\phi_i\}$ to $\{K(\boldsymbol{s}, (k_1, 0)^{\text{t}})\}$ is reasonable, and this is achieved by setting $\boldsymbol{k}_{\text{new}}$ in the neighborhood of $(k_1, 0)^{\text{t}}$. Because the critic updates $\xi_{d(k_1,0)^{\text{t}}}$ and some of the values of $|\xi_{d(k_1,0)^{\text{t}}}|$ are large, the strategy that sets $\boldsymbol{k}_{\text{new}}$ near $\boldsymbol{k}_{\text{large}}$ should work well.

(3) If $\|\boldsymbol{\xi}\|_{\boldsymbol{k}}$ for $\boldsymbol{k}$ in the neighborhood of $\boldsymbol{k}_{\text{large}}$ is not small, the NA-method often has to set $\boldsymbol{k}_{\text{new}}$ to a value that is far from $\boldsymbol{k}_{\text{large}}$ and is not related to the relationship between s_1 and s_2, which contributes to increasing the reward. On the other hand, the DA-method sets $\boldsymbol{k}_{\text{new}}$ in the $\boldsymbol{k}_{\text{large}}$ direction, so it has a higher ability to hold the relationship than the NA-method. This is clear when we consider the aforementioned case where s_2 is independent of increasing the reward. Thus, the DA-method should be superior to the NA-method. The difference in the performance between the DA-method and the NA-method was evaluated by a computer simulation, and the results are shown in Sect. 4.

By updating *IDXT* repeatedly, significant elements in $\{K(\boldsymbol{s}, \boldsymbol{k})\}$ are selected, so the accuracies of the function approximations in Eqs. (3) and (4) increase as the critic progresses in the learning. Further details of DA- and NA-methods and their algorithms are shown in Appendix A.

4. Performance Evaluation

Consider a chaos control problem for a linear combination of d_{s} logistic maps [15] defined by

$$\begin{cases} \boldsymbol{s}_{t+1} = C\boldsymbol{f}(\boldsymbol{s}_t) \\ f_d(s_{d;t}) \overset{\text{def}}{=} (a_d + u_{d;t}) s_{d;t}(1 - s_{d;t}), \end{cases} \tag{12}$$

where $d = 1, 2, \cdots, d_{\text{s}}$, $0 < s_{d;t} < 1$, $-0.1 \leq u_{d;t} \leq 0.1$, a_d is a constant, $C = [c_{dd'}]$ is a $d_{\text{s}} \times d_{\text{s}}$ matrix that defines the interaction of d_{s} logistic maps, and $\boldsymbol{f}(\boldsymbol{s}_t) \overset{\text{def}}{=} (f_1(s_{1;t}), \cdots, f_{d_{\text{s}}}(s_{d_{\text{s}};t}))^{\text{t}}$. The problem is stabilizing state $\boldsymbol{s}_t$ in Eq. (12) by adjusting control input $\boldsymbol{u}_t$, and it was solved using the actor-critic method with the method for adapting an orthonormal basis based on AIA (DA-method or NA-method). This system was stabilized by setting immediate reward r_{t+1} as

$$r_{t+1} = -(\sum_{d=1}^{d_{\text{s}}} w_{\text{s}d}(s_{d;t+1} - s_{d;t})^2 + w_{\text{u}d} u_{d;t}^2), \tag{13}$$

where $w_{\text{s}d}$ and $w_{\text{u}d}$ are given constants. The following parameters were used for the evaluations described below: $d_{\text{s}} = 1$ or 2, $w_{\text{u}1} = w_{\text{u}2} = 10$, $a_1 = 3.8$, $a_2 = 3.9$, $c_{11} = c_{22} = 1 - c$, $c_{12} = c_{21} = c$. Here, c denotes the intensity of the interaction between $s_{1;t}, \cdots, s_{d_{\text{s}};t}$. When $c = 0$, $s_{1;t}, \cdots, s_{d_{\text{s}};t}$ behave independently, and the interaction increases in proportion to c. The effects of $s_{d;t+1}$ and $u_{d;t+1}$ on r_{t+1} are proportional to $w_{\text{s}d}$ and $w_{\text{u}d}$, respectively. Index table *IDXT* was first set using Eq. (8) with $N_d = (N+1)^{1/d} - 1$ for $\forall d$, and it was modified by AIA.

First, the degree of expansion, N, required to stabilize the system with only one state ($d_{\text{s}} = 1$) was estimated by evaluating the effect of N on the convergence time. Here, the convergence time is denoted as the time it takes for the average of r_t, $\cdots$, r_{t-10^5} to become larger than -10^{-2}, and its mean and standard deviation were evaluated for various initial values of state $\boldsymbol{s}_t$. Also, $d_{\text{s}} = 1$, $w_{\text{s}1} = 1$, $c = 0$, and AIA was not used. When AIA was not used, *IDXT* was set using Eq. (8) with $N_d = (N+1)^{1/d} - 1$, and it was fixed. This is referred to as the fixed-*IDXT* from this point on. As shown in Fig. 4, the convergence time increased as N decreased, and the minimum value of N that is required

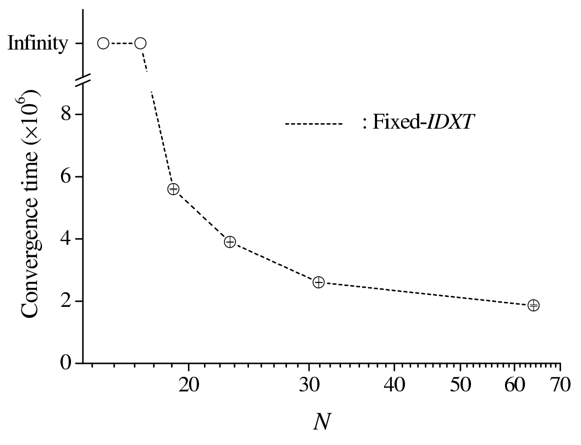


Fig. 4 Effect of degree of expansion, N, on convergence time of $s_{1;t}$ ($d_s = 1$).

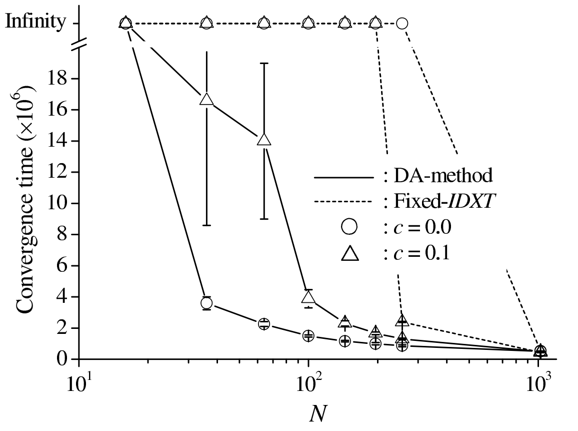


Fig. 6 Effect of DA-method on degree of expansion, N, required to stabilize $s_{1;t}$ and $s_{2;t}$.

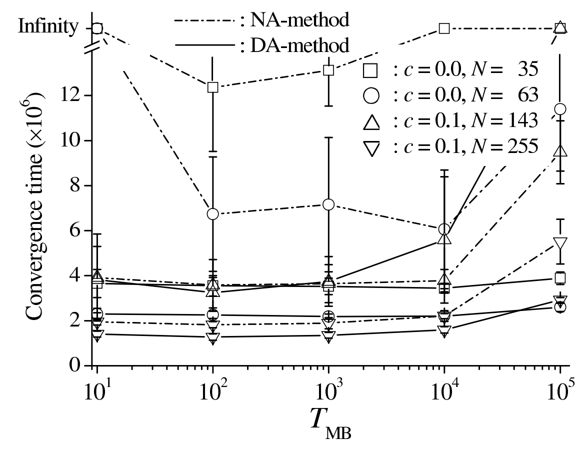


Fig. 5 Difference between DA-method and NA-method.

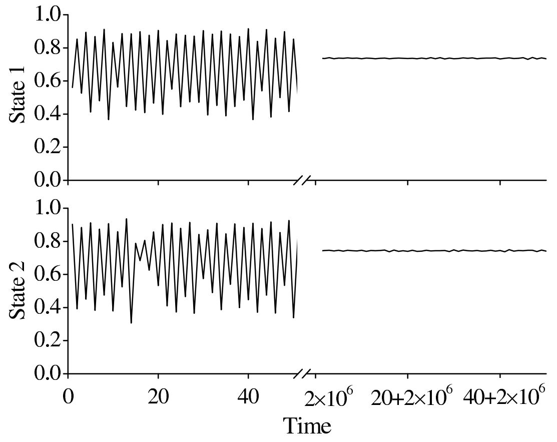


Fig. 7 Changes in $s_{1;t}$ and $s_{2;t}$ obtained using DA-method ($c = 0.1$ and $N = 255$).

to stabilize the system with $d_s = 1$ was 19.

Next, the difference between the DA-method and NA-method was evaluated. Figure 5 shows the effect of the interval of performing AIA, T_{MB}, on the convergence time when $d_s = 2$, $w_{s1} = w_{s2} = 1$, and $c = 0$ or 0.1. Here, $s_{1;t}$ and $s_{2;t}$ affect the immediate reward with the same weight ($w_{s1} = w_{s2} = 1$), $s_{1;t}$ and $s_{2;t}$ are independent of each other when $c = 0$, and $s_{1;t}$ and $s_{2;t}$ interact with each other when $c = 0.1$. As can be seen in this figure, the DA-method is superior to the NA-method from the viewpoint of the convergence time and the degree of expansion because the DA-method takes the direction of the index vector into account as described in Sect. 3. This figure also shows that T_{MB} should be 10^2 to obtain a shorter convergence time. Thus, the DA-method was used for AIA, and T_{MB} was 10^2 elsewhere in this paper.

To show the effect of the DA-method, the relationship between N and the convergence time was evaluated when $d_s = 2$, $w_{s1} = w_{s2} = 1$, and $c = 0$ or 0.1. Figure 6 shows that when the fixed-*IDXT* was used, the required value of N to stabilize the system was almost equal to the square of that in Fig. 4; that is, the required value of N approximately increased proportionally to the d_sth power of that for $d_s = 1$. When the DA-method was used, the required value of N to stabilize the system was almost equal to twice the one in Fig. 4 regardless of the interaction between the states. This means that the basis modified by the DA-method works well and that the degree of expansion N can be reduced using the DA-method. The changes in s_1 and s_2 obtained with the DA-method for $c = 0.1$ and $N = 255$ are shown in Fig. 7.

The robustness of the DA-method was evaluated using an environment with $d_s = 2$, $w_{s1} = 1$, $w_{s2} = 0$, and $c = 0$. In this environment, $s_{2;t}$ is a redundant state, and it works a disturbance noise to prevent $s_{1;t}$ from converging because $s_{2;t}$ does not affect the immediate reward ($w_{s2} = 0$) and because $s_{1;t}$ and $s_{2;t}$ are independent of each other ($c = 0$). Thus, if the DA-method is not affected by $s_{2;t}$ and works well, the degree of expansion, N, which is required to stabilize the system, should be almost the same as those in Fig. 4 because only $s_{1;t}$ is taken into account in the imme-

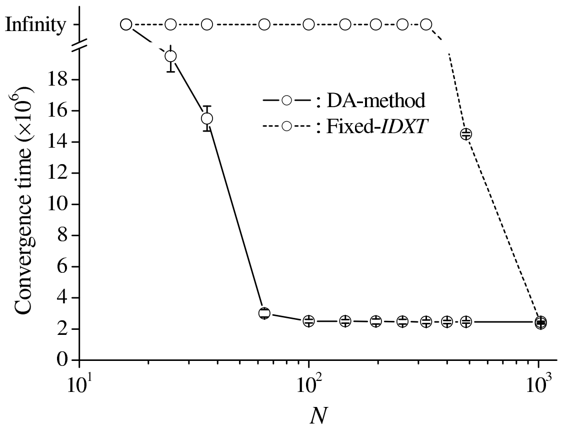


Fig. 8 Robustness of DA-method: Convergence time of $s_{1;t}$ when $s_{2;t}$ is redundant noisy state.

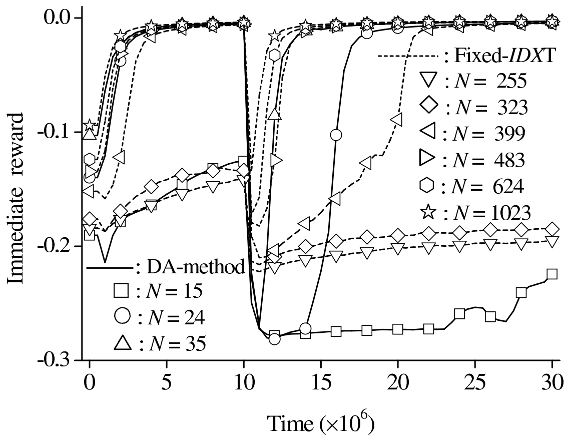


Fig. 9 Adaptability of DA-method to changes in environment.

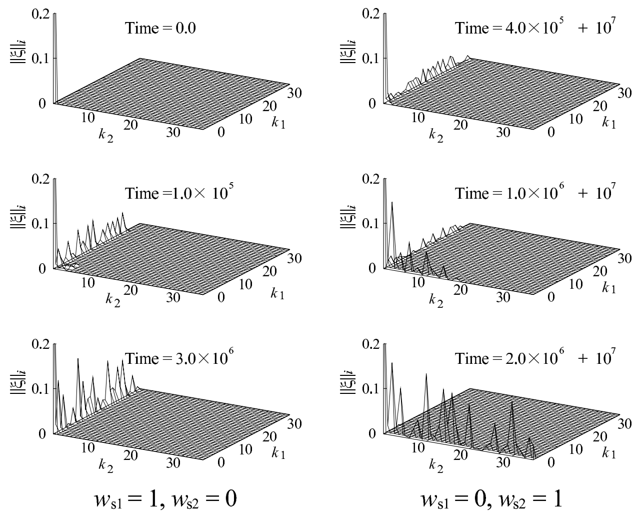


Fig. 10 Change in $\|\boldsymbol{\xi}\|_i$ in $\boldsymbol{k}$-coordinate system when DA-method was used ($N = 35$).

diate reward. As shown in Fig. 8, the required value of N when the DA-method was used was almost the same as that in Fig. 4, while that when the fixed-IDXT was used was almost the square of that in Fig. 4. This means that the DA-method eliminated the adverse effect of the redundant state and was robust, even though the fixed-*IDXT* was affected by the redundant state.

To demonstrate that the DA-method can adapt to changes in the environment, the immediate rewards were estimated for an environment in which the parameters of the immediate reward in Eq. (13) changed such that $w_{s1} = 1$ and $w_{s2} = 0$ for $0 \leq t < 10^7$ and $w_{s1} = 0$ and $w_{s2} = 1$ for $10^7 \leq t$. Here, $d_s = 2$ and $c = 0$. In this environment, s_1 and s_2 are independent of each other because $c_{12} = c_{21} = 0$. Moreover, either w_{s1} or w_{s2} is equal to 0. Thus, even though this environment has two state variables, s_1 and s_2, the reward can be practically maximized by using only one state variable, s_1 or s_2, in the same manner as in Fig. 8. This means that the value of N required for stabilizing the system in this environment should be equal to that in Fig. 4 if the DA-method can find the change in the environment. On the basis of this viewpoint, Fig. 9 shows that the actor-critic method with DA-method stabilized the system even if a small value of N that is almost equal to the value in Fig. 4 was used and that the DA-method responded to changes in the environment and modified *IDXT* according to the changes, even though the change in w_{sd} was not directly observed. However, when the fixed-*IDXT* was used, the value of N required to stabilize the system was almost the square of that when the DA-method was used.

Figures 10 and 11 show the changes in $\|\boldsymbol{\xi}\|_i$ defined in Eq. (10) and those in s_1 and s_2, respectively, when the DA-method with $N = 35$ was used in the environment for Fig. 9. Here, the relationship between index i and the coordinate corresponding to i in the $\boldsymbol{k}$-coordinate system was obtained using *IDXT*. These figures show that the DA-method modified *IDXT* on the basis of changes in the environment and that it contributed to stabilize s_1 and s_2. Therefore, we conclude that the DA-method can adaptively modify *IDXT* according to changes in the environment, thereby enabling reinforcement learning to work well.

The effect of the initial value of *IDXT* on the immediate reward is shown in Fig. 12 when the DA-method was used, $d_s = 2$, $w_{s1} = w_{s2} = 1$, and $c = 0.1$, where "Square" in the figure denotes that the initial value of *IDXT* is given by Eq. (8) with $N_1 = N_2 = (N+1)^{1/2} - 1$ and "Line" denotes that the initial value is given by Eq. (8) with $N_1 = N$ and $N_2 = 0$. As shown in this figure, the immediate rewards for both initial values increase immediately when $N = 143$. However, when $N = 63$, the immediate rewards for the "Line" shape initial value remains small, even though that for the "Square" shape initial value increases. This means that the chaos control failed when $N = 63$ and the "Line" shape initial value was used, even though it succeeded when the "Square" shape initial value was used. The

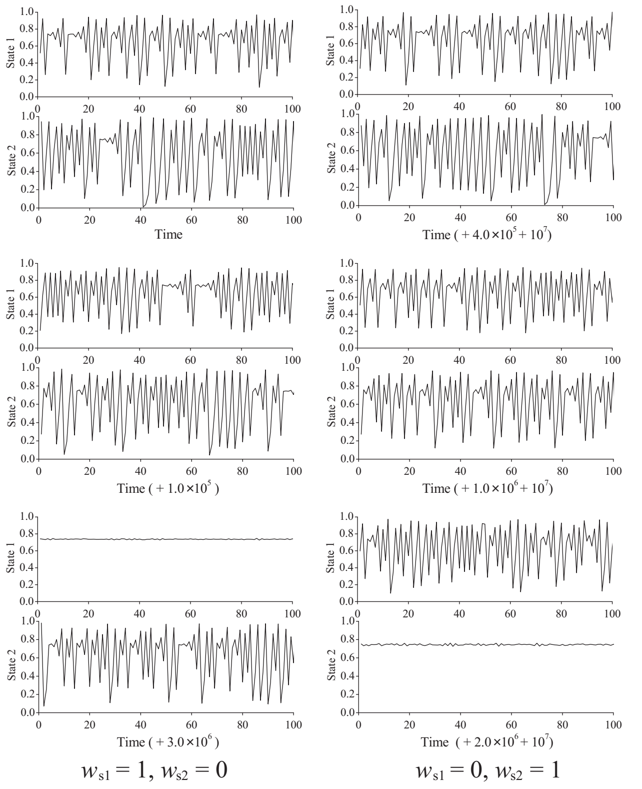


Fig. 11 Changes in $s_{1;t}$ and $s_{2;t}$ when DA-method was used ($N = 35$).

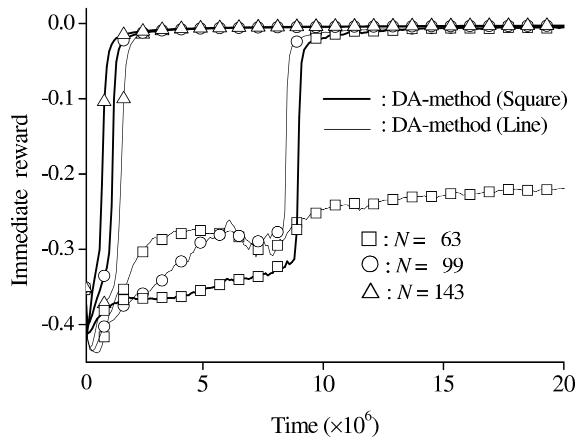


Fig. 12 Effect of initial value of *IDXT* on immediate reward when DA-method is used.

effect of *IDXT*'s initial value was higher as N decreases. This is the same problem as the initial value problem of non-linear equations, and it remains for future work to solve.

5. Conclusion

An orthonormal basis adaptation method for function approximation and its application to reinforcement learning were presented. The method first set a basis used for linear function approximation of a control function to an orthonormal basis. Next, it replaces the basis elements with small activities with other candidate elements as learning progresses. As this replacement is repeated, the number of basis elements with large activities increases. The method presented in this paper modifies the basis only using the activities of the basis elements. Even though it does not compute any statistical properties such as eigen values or eigen vectors, it has not only small calculation cost but also high performance; it eliminated the adverse effects of redundant noisy states, reduced the number of basis elements required to perform reinforcement learning, and modified the basis while holding the orthonormality in accordance with changes in the environment so that reinforcement learning was enhanced.

Thus, we can apply the orthonormal basis adaptation method to hybrid reinforcement learning to improve the performance under a restriction of the number of trials for reinforcement learning. Moreover, a combination of the orthonormal basis adaptation method and the state space compression method based on multivariate analysis [14] should be effective to control various environments with a higher-dimensional state space than those evaluated in this paper. These two studies should contribute to solving the problems of the curse of dimensionality and a large number of trials in actual systems. These will be reported in the near future.

References

[1] R. S. Sutton and A. G. Barto, Reinforcement Learning, MIT Press, USA, 1998.

[2] Baird, L. C. and Klopf, A. H., "Reinforcement learning with high-dimensional, continuous actions," Technical Report WL-TR-93-1147, 1993.

[3] W. T. B. Uther and M. M. Veloso, "Tree based discretization for continuous state space reinforcement learning,", Proceedings of AAAI-98, pp. 769–774, July, 1998.

[4] K. Samejima and T. Omori, "Adaptive internal state space construction method for reinforcement learning of a real-world agent," Neural Networks, vol. 12, no. , pp. 1143–1155, 1999.

[5] I. Menache and S. Mannor and N.Shimkin, "Basis function adaptation in temporal difference reinforcement learning," Annals of Operations Research, vol. 134, no. 1, pp. 215–238, Feb. 2005.

[6] S. Mahadevan and M. Maggioni, "Value function approximation using diffusion wavelets and Laplacian eigenfunctions," Neural Information Processing Systems (NIPS) MIT Press, 2006.

[7] J. Johns and S. Mahadevan, "Constructing basis functions from directed graphs for value function approximation," Proceedings of the 24th international conference on Machine learning, pp. 385–392, 2007.

[8] T. Kondo and I. Koji, "A Reinforcement Learning with Evolutionary State Recruitment Strategy for Autonomous Mobile Robots Control," Robotics and Autonomous Systems, vol. 46, no. 2, pp. 111–124, 2004.

[9] P. M. Mills and M. O. Tade and A. Y. Zomaya, "Identification and control using a hybrid reinforcement learning sys-

i \ k_d	k_1	k_2
0	0	0
1	1	0
2	0	1
3	1	1
4	0	2
5	1	2

(a) *IDXT*

i' \ k_d	k_1	k_2
0	0	0
1	1	2
2	1	1
3	1	0
4	0	1
5	0	2

(b) *IDXTS* of (a)

i'	i
0	0
1	5
2	3
3	1
4	2
5	4

(c) ***IdxS*** of (a)

Fig. A·1 Examples of index table and sorted index table.

tem," International Journal of Computer Simulation, vol. 5, no. 2, pp. 109–126, 1995.

[10] S. Liu and G. P. Henze, "Experimental analysis of simulated reinforcement learning control for active and passive building thermal storage inventory," Energy and Buildings, vol. 38, no. 2, pp. 142-147, Feb. 2006.

[11] A. Isidori, Nonlinear Control Systems 3rd ed., Springer-Verlag, USA, 1995.

[12] K. Glover, D. J. N. Limebeer, J. C. Doyles, E. M. Kasenally, and M. G. Safonov: A characterization of all solutions to the four block general distance problem, *SIAM Journal on Control and Optimization*, **29**-2, 283/324, (1991)

[13] G. F. Franklin and J. D. Powell: Digital Control and Dynamic Systems, Addison-Wesley, USA (1980)

[14] H. Satoh, "A state space compression method based on multivariate analysis for reinforcement learning in high-dimensional continuous state spaces," IEICE Trans. Fundamentals, vol. E89-A, no. 8, pp. 2181–2191, Aug. 2006.

[15] E. Ott, Chaos in Dynamical Systems Second edition, Cambridge, UK, 2002.

Appendix A: Algorithm of Activity-Oriented Index Allocation

Consider an index table, *IDXT*, and $\|\boldsymbol{\xi}\|_i$. Let *IDXTS* be an $(N+1) \times d_{\rm s}$ matrix that represents an index table whose rows are sorted by $\|\boldsymbol{\xi}\|_i$, and let $\boldsymbol{IdxTS}_{i'}{}^{\rm t} = (IdxTS_{i'1}, \cdots, IdxTS_{i'd_{\rm s}})$ be the i' row of *IDXTS*. Let $\boldsymbol{IdxS}$ be the $(N+1)$ vector whose i' element, $IdxS_{i'}$, represents the index of the basis, i, corresponding to the row of *IDXTS*. Here, $\boldsymbol{IdxTS}_0$ is always set to the 0th row of *IDXT*, and $\|\boldsymbol{\xi}\|_{IdxS_{i'}} > \|\boldsymbol{\xi}\|_{IdxS_{i'+1}}$ for $i' > 0$. Examples of *IDXT*, *IDXTS*, and $\boldsymbol{IdxS}$ are shown in Fig. A·1, in which the value of $\|\boldsymbol{\xi}\|_i$ related to index i by *IDXT* is ordered as $\|\boldsymbol{\xi}\|_5 > \|\boldsymbol{\xi}\|_3 > \|\boldsymbol{\xi}\|_1 > \|\boldsymbol{\xi}\|_2 > \|\boldsymbol{\xi}\|_4$. Let $n_{\rm MB}$ be the number of elements in the basis to be replaced for each replacement and $\boldsymbol{k}_{\rm free}$ be a candidate for the new index vector. The activity-oriented index allocation methods (DA- and NA-methods) described in Sect. 3 are performed by Algorithm 1.

Algorithm 1: Activity-oriented index allocation method (DA- and NA-methods)

(2-1) Set *IDXTS* from *IDXT*.

(2-2) Set $i' = N - n_{\rm MB} + 1$ and $j' = 0$.

(2-3) Set $\boldsymbol{k}_{\rm free}$ using Algorithm 2 for the DA-method or Algorithm 3 for the NA-method.

(2-4) If $\boldsymbol{k}_{\rm free} \notin \{\boldsymbol{IdxTS}_0, \cdots, \boldsymbol{IdxTS}_{i'-1}\}$, perform the following steps:

  (1) Set $\boldsymbol{k}_{\rm free}$ to $IdxS_{i'}$th row of *IDXT*.

  (2) Set $\boldsymbol{k}_{\rm free}$ to i'th row of *IDXTS*.

  (3) Set $\xi_{IdxS_{i'}}$ and $\eta_{IdxS_{i'}}$ to 0.

  (4) Set $i' = i' + 1$.

(2-5) Set $j' = j' + 1$.

(2-6) If $j' = N - n_{\rm MB} + 1$, set $j' = 0$.

(2-7) If $i' \leq N$, go to Step (2-3). Otherwise, go to Step (2-8).

(2-8) Set $\phi_i(\boldsymbol{s})$ by Eq. (7) for $0 \leq i \leq N$ based on *IDXT*.

The algorithms used for Step (2-3) are described below.

Algorithm 2: Setting of $\boldsymbol{k}_{\rm free}$ for the DA-method

A candidate for a new index vector, $\boldsymbol{k}_{\rm free}$, is set in the $\boldsymbol{IdxTS}_{j'}$ direction in the $\boldsymbol{k}$-coordinate system so that $\boldsymbol{k}_{\rm free} \in \boldsymbol{\mathcal{D}}_{\mathbf{k}}$, $\boldsymbol{k}_{\rm free}$ is as close as possible to $\mathbf{0}$, and $\boldsymbol{k}_{\rm free} \notin \{\boldsymbol{IdxTS}_0, \cdots, \boldsymbol{IdxTS}_{i'-1}\}$, where $\boldsymbol{\mathcal{D}}_{\mathbf{k}} \overset{\rm def}{=} \{\boldsymbol{k} | dist \times (IdxTS_{j'd} - disp/2) \leq k_d \leq dist \times (IdxTS_{j'd} + disp/2), \forall dist > 0, k_d \geq 0$, and $1 \leq d \leq d_{\rm s}\}$ is the domain for searching for $\boldsymbol{k}_{\rm free}$ in the $\boldsymbol{k}$-coordinate system, and $disp$ is a constant that defines the extent of $\boldsymbol{\mathcal{D}}_{\mathbf{k}}$. Note that Algorithm 3 is used when $\boldsymbol{IdxTS}_{j'} = \mathbf{0}$.

Algorithm 3: Setting of $\boldsymbol{k}_{\rm free}$ for the NA-method

A candidate for a new index vector, $\boldsymbol{k}_{\rm free}$, is set randomly in $\boldsymbol{\mathcal{D}}_{\mathbf{k}}$, where $\boldsymbol{\mathcal{D}}_{\mathbf{k}} \overset{\rm def}{=} \{\boldsymbol{k} | IdxTS_{j'd} - disp/2 \leq k_d \leq IdxTS_{j'd} + disp/2, k_d \geq 0$, and $1 \leq d \leq d_{\rm s}\}$ is the domain for searching for $\boldsymbol{k}_{\rm free}$ in the $\boldsymbol{k}$-coordinate system, and $disp$ is a constant that defines the extent of $\boldsymbol{\mathcal{D}}_{\mathbf{k}}$.

Here, $disp$ determines the area of the search space shown in Fig. 2 and is set to 0.4 for the DA-method and 2 for the NA-method for the performance evaluation in Sect. 4.