TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 03/18 pp519-526
Volume 9, Number 5, October 2004

Advanced Credit-Assignment CMAC Algorithm for Robust Self-Learning and Self-Maintenance Machine*

ZHANG Lei (张 蕾)**, LEE Jay†, CAO Qixin (曹其新), WANG Lei (王 磊)

Research Institute of Robotics, Shanghai Jiao Tong University, Shanghai 200030, China;
Research Center of Intelligent Maintenance Systems, University of Wisconsin-Milwaukee, WI 53224, USA

Abstract: Smart machine necessitates self-learning capabilities to assess its own performance and predict its behavior. To achieve self-maintenance intelligence, robust and fast learning algorithms need to be embedded in machine for real-time decision. This paper presents a credit-assignment cerebellar model articulation controller (CA-CMAC) algorithm to reduce learning interference in machine learning. The developed algorithms on credit matrix and the credit correlation matrix are presented. The error of the training sample distributed to the activated memory cell is proportional to the cell's credibility, which is determined by its activated times. The convergence processes of CA-CMAC in cyclic learning are further analyzed with two convergence theorems. In addition, simulation results on the inverse kinematics of 2-degree-of-freedom planar robot arm are used to prove the convergence theorems and show that CA-CMAC converges faster than conventional machine learning.

Key words: cerebellar model articulation controller; machine learning; self-maintenance machine; self-learning

Introduction

Smart machine necessitates self-monitoring, self-diagnosing, and self-maintenance capabilities. The key challenge is to embed fast learning algorithms in machine controller to enable real-time behavior assessment and prognostic intelligence. Many neural networks-based algorithms have been developed to achieve this objective in the past decades. Among these developed tools, the cerebellar model articulation controller (CMAC) network developed by Albus[1,2] has the unique advantages of local generalization, extremely fast learning and easy implementation in software and hardware, so it is considered as a good alternative to the backpropagation method[3]. CMAC has been widely used in many fields, such as function approximation[4], chaotic time series prediction[5], especially in robotic control[6-8]. Wong and Sideris[9] discussed the learning convergence of CMAC, and pointed out that the CMAC learning algorithm in cyclic learning is equivalent to the Gauss Siedel iterative scheme of linear system. However, only a special case was considered in the paper where the learning rate of CMAC was one and the training samples were noiseless. Yao and Zhang[10] extended Wong et al.'s results to the case where the learning rate is other than one and discussed the learning convergence of CMAC when the training samples have noises. They proved that the CMAC learning scheme converges if and only if the learning rate is chosen from (0, 2). Lin and Chiang[11] also proved that the memory contents of CMAC either with or without hash mapping will converge to a limit cycle providing that the learning rate is

Received: 2004-06-24
* Supported by the National Natural Science Foundation of China (No. 50128504)
** To whom correspondence should be addressed.
E-mail: zhanglei@sjtu.edu.cn; Tel: 86-21-62932711

in (0, 2). This conclusion is the same as Yao et al.' s. Lee and Kramer[12] introduced an innovative concept in using CMAC for machine performance degradation assessment and presented some pioneer work with experiments. They further presented the advanced concepts in using the developed self-learning capabilities for remote monitoring and prognostics[13,14].

There are two basic learning schemes in CMAC: cyclic learning and random learning. In both of these two schemes, there exists a problem termed as "learning interference" which means that training of subsequent samples will destroy the precision of previous ones. Learning interference is one of the important reasons decreasing convergence speed of CMAC. To reduce learning interference, Thompson[15] suggested a method named as neighborhood sequential training. Sayil and Lee[16] developed a hybrid maximum error with neighborhood training algorithm. But in neighborhood training algorithm, training points that lie outside of the neighborhood of the previous training points are chosen. It is usually impossible that the two methods are applied practically if the function to be learned is unknown.

In Ref. [17], Su et al. suggested a credit-assignment CMAC (CA-CMAC) algorithm to speed up the convergence of CMAC, but they did not make any analyses on the convergence property of CA-CMAC. In this paper, we propose a more general CA-CMAC algorithm. We further present the concepts of the credit matrix and the credit correlation matrix, and then prove that the conventional CMAC algorithm is a special case of the proposed CA-CMAC algorithm. Furthermore, the convergence properties of CA-CMAC in cyclic learning are investigated while the training samples without or with noises, respectively. The convergence theorems are obtained, which are significant as a guidance to choose the learning rate in CA-CMAC practical applications. Finally, simulations are carried out to compare the convergence performance of CA-CMAC with that of conventional CMAC.

1 Credit-Assignment CMAC Algorithm

CMAC can be considered as an associative memory network, which performs two subsequent mappings: $f: \boldsymbol{X} \to \boldsymbol{A}, h: \boldsymbol{A} \to \boldsymbol{P}$, where $\boldsymbol{X}$ is an M-dimensional input space. $\boldsymbol{A}$ is an N-dimensional association cell vector which contains g nonzero elements. g is the generalization parameter. $\boldsymbol{P}$ is a one-dimensional output space. In the first mapping, the point $\boldsymbol{X}_k$ in the input space is mapped into a binary associative vector $\boldsymbol{A}_k$ whose elements are defined as Eq. (1). In the second mapping, the network output is calculated as the scalar product of $\boldsymbol{A}_k$ and the weight vector $\boldsymbol{W}$, as shown in Eq. (2). The update rule to the weights is shown in Eq. (3). For the simplicity of analysis on the convergence, hash coding is not considered here.

$$a_{k,j} = \begin{cases} 1, \text{ if the } j\text{-th element is activated} \\ \quad \text{by the } k\text{-th sample;} \qquad\qquad 1 \le j \le N \\ 0, \text{ otherwise.} \end{cases} \tag{1}$$

$$Y_{\mathrm{r},k} = \boldsymbol{A}_k^{\mathrm{T}}\, \mathrm{g}\boldsymbol{W} = \sum_{j=1}^{N} a_{k,j} w_j \tag{2}$$

$$w_j(t+1) = w_j(t) + \Delta w_j(t) =$$
$$w_j(t) + \frac{b\ a_{k,j}}{g}(Y_{\mathrm{d},k} - \sum_{j=1}^{N} a_{k,j} w_j(t)) \tag{3}$$

where k is the k-th sample, $Y_{\mathrm{r},k}$ the real output of the k-th sample, w_j the j-th element in the weight vector $\boldsymbol{W}$, t the t-th cycle, $\hat{a}$ the learning rate, and $Y_{\mathrm{d},k}$ the desired output of the k-th sample.

From Eq. (3), it can be seen that all addressed memory cells get equal shares for error correction of the sample ($\boldsymbol{X}_k$, $Y_{\mathrm{d},k}$) in the t-th cycle. This will result in that previous learned information is corrupted due to learning interference. In fact, the memory cells activated by ($\boldsymbol{X}_k$, $Y_{\mathrm{d},k}$) may have different learning histories, and thus have the different credibility. Based on the concept of credit assignment, we assign the credibility to each memory cell, which is the inverse of the cell' s activated times. The error distribution is proportional to the credibility. The modified update rule to the weights can be written as

$$w_j(t+1) = w_j(t) + b\ a_{k,j} \frac{1/f(j)}{\sum_l (1/f_k(l))}(Y_{\mathrm{d},k} - \sum_{j=1}^{N} a_{k,j} w_j) \tag{4}$$

where $f(j)$ is the activated times of the j-th memory cell. $f_k(l)$ is the activated times of the memory cell activated by the k-th sample. To prevent dividing by zero, the minimum value of the activated times of the memory cell is set to 1 firstly. The more times the memory cell has been activated, the more accurate the stored weight is. Therefore, the equal share of error correcting as $1/g$ in Eq. (3) is replaced by $(1/f(j))/\sum_{l}(1/f_k(l))$ in Eq. (4). With this modification, the error of the training sample can be proportionally distributed into the activated memory cell based on its credibility. It is obvious that the modified error distribution is more reasonable.

2 The Convergence Property of CA-CMAC

Wong and Sideris[9] had proposed another description of the conventional CMAC algorithm to analyze the learning convergence. In this section, we will describe the learning process of CA-CMAC in the similar way.

Suppose that the training samples are ($\boldsymbol{X}_i$, $Y_{\mathrm{d},i}$) (i=1,2,$\cdots$, n), and n is the number of training samples. Consider the t-th cycle of the k-th training sample. The output error $d_k^{(t)} = Y_{\mathrm{d},k} - \boldsymbol{A}_k^{\mathrm{T}}\boldsymbol{W}^{(t)}$. Let $g_k = \sum_{l}(1/f_k(l))$ and define the unit correction as $d_k^{(t)}/g_k$. From Eq. (4), correction for the weight in the memory cell activated by the k-th sample is $\dfrac{1/f_k(l)d_k^{(t)}}{g_k}$. The correction may affect the outputs of other training samples. At this time, the output of the i-th training sample becomes $\boldsymbol{A}_i^{\mathrm{T}}\boldsymbol{W}^{(t)} + d_{ik}d_k^{(t)}/g_k$, where $d_{ik} = \sum_{l}(1/f_{ik}(l))$, $f_{ik}(l)$ is the activated times of the memory cell which is activated by both the i-th sample and the k-th sample. Obviously, $d_{ik} = d_{ki}$, $d_{kk} = \sum_{l}(1/f_k(l)) = g_k$. If the initial weights are set to zeros, after t cycles, the accumulated unit correction of the k-th sample is $D_k' = \sum_{t} d_k^{(t)}/g_k$. Then the output of the i-th sample can be expressed as $Y_{\mathrm{r},i} = \sum_{k=1}^{n} d_{ik}D_k'$. When the CA-CMAC learning converges, i.e., $\boldsymbol{d}_k^{(t)}$ goes to zero, D_k' will converge to a constant. Therefore, the convergence of CA-CMAC algorithm is equivalent to the convergence of D_k'. Let $\boldsymbol{D} = (d_{ik})$ $(i,k = 1,2,\cdots,n)$, $D' = [D_1', D_2', \cdots, D_n']^{\mathrm{T}}$. Expressed in a matrix form, the learning convergence is equivalent to find the solutions of the linear system,

$$\boldsymbol{D}D' = \boldsymbol{Y} \tag{5}$$

where $\boldsymbol{Y} = [Y_{\mathrm{d},1}, Y_{\mathrm{d},2}, \cdots, Y_{\mathrm{d},n}]^{\mathrm{T}}$.

Let $\boldsymbol{A} = [\boldsymbol{A}_1, \boldsymbol{A}_2, \cdots, \boldsymbol{A}_n]$ and define a matrix $\boldsymbol{F}$,

$$\boldsymbol{F} = \begin{bmatrix} 1/f(1) & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1/f(2) & 0 & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & 1/f(N-2) & 0 & 0 \\ 0 & \cdots & 0 & 0 & 1/f(N-1) & 0 \\ 0 & \cdots & 0 & 0 & 0 & 1/f(N) \end{bmatrix}_{N\times N}.$$

The diagonal elements of $\boldsymbol{F}$ represent the credibility of memory cells, so we call $\boldsymbol{F}$ the credit matrix. From the definition of $\boldsymbol{D}$, we can get

$$\boldsymbol{D} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{A} \tag{6}$$

We call $\boldsymbol{D}$ the credit correlation matrix. $\boldsymbol{D}$ has many good properties, which can be described in Theorem 1.

Theorem 1 The credit correlation matrix $\boldsymbol{D}$ has following properties: a) $\boldsymbol{D}$ is a real symmetric matrix. All elements are non-negative integers, and the diagonal elements are g_k (k = 1, 2, $\cdots$, n). b) $\boldsymbol{D}$ is a positive semidefinite matrix.

Proof Property a) can be obtained easily from the definition of $\boldsymbol{D}$. Property b) is proved as follows: $\exists\, \boldsymbol{X} \neq \boldsymbol{0}$, quadratic form $f(\boldsymbol{X}) = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{X} = \boldsymbol{X}^{\mathrm{T}}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{A})\boldsymbol{X} = (\boldsymbol{A}\boldsymbol{X})^{\mathrm{T}}\boldsymbol{F}(\boldsymbol{A}\boldsymbol{X}) \geq 0$, so $\boldsymbol{D}$ is positive semidefinite.

Then Eq. (5) can be written as

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{A}D' = \boldsymbol{Y} \tag{7}$$

If $f(i) = 1$ $(i = 1,2,\cdots,N)$, $\boldsymbol{F}$ will become the unit matrix and g_k (k = 1, 2, $\cdots$, n) equals to g. Then Eq. (7) becomes

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}D = \boldsymbol{Y} \tag{8}$$

where $D = [D_1, D_2, \cdots, D_n]^{\mathrm{T}}$, $D_k = \sum_{t} d_k^{(t)}/g$ (k=1,2, $\cdots$, n). Equation (8) is no other than the equation of the linear system described by Wong and Sideris in the conventional CMAC algorithm[9]. So the conventional CMAC algorithm is just a special case of CA-CMAC, in which all the memory cells have the same activated times, and thus have the same credibility.

In this paper, only cyclic learning is considered, where the training samples are presented to the CA-CMAC in a cyclic fashion and weights are updated after each presentation. The convergence properties of CA-CMAC are investigated in two cases: the training samples with or without noises.

2.1 Learning convergence of CA-CMAC without noises

In cyclic learning, the update rule of the *i*-th sample in the *t*-th cycle can be written as

$$D_i'^{(t+1)} = D_i'^{(t)} + b\,(Y_{\mathrm{d},i} - \sum_{j=1}^{i-1} d_{ij} D_j'^{(t+1)} - \sum_{j=i}^{n} d_{ij} D_j'^{(t)}) / g_i \tag{9}$$

Define a matrix

$$\boldsymbol{G} = \begin{bmatrix} g_1 & 0 & 0 & \mathrm{L} & 0 & 0 \\ 0 & g_2 & 0 & \mathrm{L} & 0 & 0 \\ \mathrm{L} & \mathrm{L} & \mathrm{L} & \mathrm{L} & \mathrm{L} & \mathrm{L} \\ 0 & \mathrm{L} & 0 & g_{n-2} & 0 & 0 \\ 0 & \mathrm{L} & 0 & 0 & g_{n-1} & 0 \\ 0 & 0 & 0 & 0 & 0 & g_n \end{bmatrix}_{n\times n}.$$

$\boldsymbol{D}$ can be written as $\boldsymbol{D} = \boldsymbol{L}+\boldsymbol{G}+\boldsymbol{U}$. $\boldsymbol{L}$ and $\boldsymbol{U}$ are the lower and upper off-diagonal parts of $\boldsymbol{D}$, respectively. $\boldsymbol{G}$ is the diagonal part of $\boldsymbol{D}$. Then Eq. (9) becomes

$$\boldsymbol{G}D'^{(t+1)} = \boldsymbol{G}D'^{(t)} + b\,(\boldsymbol{Y} - \boldsymbol{L}D'^{(t+1)} - (\boldsymbol{G}+\boldsymbol{U})D'^{(t)}) \tag{10}$$

i.e., $D'^{(t+1)} =$

$$(\boldsymbol{G} + b\,\boldsymbol{L})^{-1}[(1-b)\boldsymbol{G} - b\boldsymbol{U}]D'^{(t)} + b\,(\boldsymbol{G} + b\,\boldsymbol{L})^{-1}\boldsymbol{Y} \tag{11}$$

Equation (11) is no other than the successive over relaxation (SOR) scheme of the linear system (Eq. (5)) with b being the relaxation factor. From the iteration theories of the linear equations[18], we get Lemmas 1 and 2.

Lemma 1 The necessary condition for the convergence of SOR scheme of the linear system $\boldsymbol{Ax}=\boldsymbol{b}$ that is the relaxation factor w fulfils $0<w<2$.

Lemma 2 If $\boldsymbol{A}$ is a positive definite matrix, when the relaxation factor w fulfils $0<w<2$, SOR scheme of the linear system $\boldsymbol{Ax} = \boldsymbol{b}$ converges.

Then the following theorem is obtained.

Theorem 2 The necessary convergence condition of CA-CMAC in cyclic learning when the training samples are noiseless is that the learning rate b fulfils $0<b<2$. Specially, when $\boldsymbol{D}$ is a positive definite matrix, $0<b<2$ becomes the sufficient and necessary condition.

Proof It can be easily obtained from Lemma 1 and Lemma 2.

2.2 Learning convergence of CA-CMAC with noises

Assuming when the *i*-th sample is presented to CA-CMAC, a noise term h_i is added to the desired output $Y_{\mathrm{d},i}$ and h_i are independent and identically distributed random variables with $E(h_i)=0$ and $E(h_i^2)=s^2$, we can prove the following theorem.

Theorem 3 When the training samples have noises, assuming $\sum_t^{\infty} b_t = \infty$, $\sum_t^{\infty} b_t^2 < \infty$, and $0<\hat{a}_t<2$, CA-CMAC algorithm converges with a probability of one in cyclic learning if b_t decreases dynamically. In other words, $\lim_{t\to\infty} E(\|D'^{(t)} - D'^{*}\|^2) = 0$, where $D'^{*} = \boldsymbol{D}^{-1}\boldsymbol{Y}$.

To prove Theorem 3, we first introduce the following lemmas.

Lemma 3 Under the assumptions of Theorem 3, $E(D'^{(t)})$ converges.

Proof In the case with noises, the update rule of the *t*-th cycle can be written as

$$D_i'^{(t+1)} = D_i'^{(t)} + b_t(Y_{\mathrm{d},i} + h_i - \sum_{j=1}^{i-1} d_{ij} D_j'^{(t+1)} - \sum_{j=i}^{n} d_{ij} D_j'^{(t)}) / g_i \tag{12}$$

Like Eq. (9), Eq. (12) can be rewritten in a matrix form,

$$D'^{(t+1)} = (\boldsymbol{G} + b_t\boldsymbol{L})^{-1}[(1-b_t)\boldsymbol{G} - b_t\boldsymbol{U}]D'^{(t)} + b_t(\boldsymbol{G} + b_t\boldsymbol{L})^{-1}(\boldsymbol{Y} + h) \tag{13}$$

where $h = (h_1, h_2, \mathrm{L}\ , h_n)^{\mathrm{T}}$. Taking expectations from both sides of Eq. (13), we obtain

$$E(D'^{(t+1)}) = (\boldsymbol{G} + b_t\boldsymbol{L})^{-1}[(1-b_t)\boldsymbol{G} - b_t\boldsymbol{U}]E(D'^{(t)}) + b_t(\boldsymbol{G} + b_t\boldsymbol{L})^{-1}\boldsymbol{Y} \tag{14}$$

Equation (14) is another form of SOR scheme. Under the assumptions of Theorem 3, we have $0<b_t<2$. Following Theorem 4.1(b) of Ref. [19],

$E(D'^{(t)})$ converges.

Lemma 4 Suppose that $\{z_n\}$ is a sequence of non-negative real numbers satisfying

$$z_{n+1} \le (1-g_n)z_n + \mathrm{O}(g_n^2), \quad z_1 = r_0 \qquad (15)$$

where r_0 is an arbitrary non-negative real number and $0<g_n<1$, $\sum_n^{\infty} g_n = \infty$, $\sum_n^{\infty} g_n^2 < \infty$, then we have $\lim_{n\to\infty} z_n = 0$.

The proof can be referred in Ref. [10].

Note that Eq. (12) can be written as

$$D'^{(t)}_{i+1} = D'^{(t)}_i + b_t \boldsymbol{S}_i (Y_{\mathrm{d},i} + h_i - \boldsymbol{D}_i^{\mathrm{T}} D'^{(t)}_i)/\boldsymbol{g}_i \qquad (16)$$

where $D'^{(t)}_{i+1}$ is the D' vector after the i-th presentation at the t-th cycle. (Here, we adopt the convention that $D'^{(t)}_{n+1} = D'^{(t+1)}_1$). $\boldsymbol{S}_i$ is an n-dimensional vector with the i-th element being 1 and others being 0. $\boldsymbol{D}_i^{\mathrm{T}}$ is the i-th row of matrix $\boldsymbol{D}$. We redefine $\boldsymbol{S}_i/g_i$ as $\boldsymbol{S}_i$, and then Eq. (16) can be rewritten as

$$D'^{(t)}_{i+1} = (1 - b_t \boldsymbol{S}_i \boldsymbol{D}_i^{\mathrm{T}}) D'^{(t)}_i + b_t \boldsymbol{S}_i (Y_{\mathrm{d},i} + h_i) \qquad (17)$$

By repeatedly applying Eq. (17) to all the samples in the training cycle, we get

$$D'^{(t+1)}_1 = \prod_{i=1}^{n} (1 - b_t \boldsymbol{S}_i \boldsymbol{D}_i^{\mathrm{T}}) D'^{(t)}_1 + b_t \sum_{i=1}^{n} \boldsymbol{S}_i (Y_{\mathrm{d},i} + h_i) \prod_{j=i+1}^{n} (1 - b_t \boldsymbol{S}_j \boldsymbol{D}_i^{\mathrm{T}}) = (1 - b_t \sum_{i=1}^{n} \boldsymbol{S}_i \boldsymbol{D}_i^{\mathrm{T}}) D'^{(t)}_1 + b_t \sum_{i=1}^{n} \boldsymbol{S}_i (Y_{\mathrm{d},i} + h_i) + b_t^2 \boldsymbol{P}(b_t) D'^{(t)}_1 + b_t^2 \boldsymbol{R}_0(b_t) + b_t^2 \sum_{i=1}^{n} \boldsymbol{R}_i(b_t) h_i \qquad (18)$$

where $\boldsymbol{R}_0(\hat{a}_t)$, $\boldsymbol{R}_1(\hat{a}_t)$, $\cdots$, $\boldsymbol{R}_n(\hat{a}_t)$ are n-dimensional vectors, $\boldsymbol{P}(\hat{a}_t)$ is an $n\times n$ matrix, and all their entries are bounded when $\hat{a}_t$ approaches zero. Let $\boldsymbol{R}(\hat{a}_t) = (\boldsymbol{R}_1(\hat{a}_t), \cdots, \boldsymbol{R}_n(\hat{a}_t))$. Define a matrix

$$\boldsymbol{G}^{-1} = \begin{bmatrix} 1/g_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1/g_2 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & 1/g_{n-2} & 0 & 0 \\ 0 & \cdots & 0 & 0 & 1/g_{n-1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/g_n \end{bmatrix}_{n\times n}.$$

Equation (18) can be rewritten in a simple matrix form

$$D'^{(t+1)} = D'^{(t)} + b_t \boldsymbol{G}^{-1}(\boldsymbol{Y} + h - \boldsymbol{D}D'^{(t)}) + b_t^2 \boldsymbol{P}(b_t) D'^{(t)} + b_t^2 \boldsymbol{R}_0(b_t) + b_t^2 \boldsymbol{R}(b_t) h \qquad (19)$$

Proof of Theorem 3

Define $d_t = D'^{(t)} - D'^{*}$. Subtract D'^{*} from both sides of Eq. (19),

$$d_{t+1} = (\boldsymbol{I} - b_t \boldsymbol{G}^{-1}\boldsymbol{D}) d_t + b_t \boldsymbol{G}^{-1} h + b_t^2 \boldsymbol{P}(b_t) d_t + b_t^2 \boldsymbol{P}(b_t) D'^{*} + b_t^2 \boldsymbol{R}_0(b_t) + b_t^2 \boldsymbol{R}(b_t) h \qquad (20)$$

We redefine $b_t^2 \boldsymbol{P}(b_t) D'^{*} + b_t^2 \boldsymbol{R}_0(b_t)$ as $b_t^2 \boldsymbol{R}_0(b_t)$, and thus Eq. (20) becomes

$$d_{t+1} = (\boldsymbol{I} - b_t \boldsymbol{G}^{-1}\boldsymbol{D}) d_t + b_t \boldsymbol{G}^{-1} h + b_t^2 \boldsymbol{P}(b_t) d_t + b_t^2 \boldsymbol{R}_0(b_t) + b_t^2 \boldsymbol{R}(b_t) \qquad (21)$$

Squaring both sides of Eq. (21) and taking conditional expectations, we get

$$\begin{aligned} E(\|d_{t+1}\|^2 \mid D'^{(1)}, \cdots, D'^{(t)}) = {} & d_t^{\mathrm{T}} (\boldsymbol{I} - b_t \boldsymbol{G}^{-1}\boldsymbol{D})(\boldsymbol{I} - b_t \boldsymbol{G}^{-1}\boldsymbol{D}) d_t + b_t^2 d_t^{\mathrm{T}} (\boldsymbol{I} - b_t \boldsymbol{G}^{-1}\boldsymbol{D}) \boldsymbol{P}(b_t) d_t + \\ & b_t^2 d_t^{\mathrm{T}} (\boldsymbol{I} - b_t \boldsymbol{G}^{-1}\boldsymbol{D}) \boldsymbol{R}_0(b_t) + b_t^2 E h^{\mathrm{T}} \boldsymbol{G}^{-1} \boldsymbol{G}^{-1} h + b_t^3 E h^{\mathrm{T}} \boldsymbol{G}^{-1} \boldsymbol{R}(b_t) h + \\ & b_t^2 d_t^{\mathrm{T}} \boldsymbol{P}^{\mathrm{T}}(b_t) (\boldsymbol{I} - b_t \boldsymbol{G}^{-1}\boldsymbol{D}) d_t + b_t^4 d_t^{\mathrm{T}} \boldsymbol{P}^{\mathrm{T}}(b_t) \boldsymbol{P}(b_t) d_t + b_t^4 d_t^{\mathrm{T}} \boldsymbol{P}^{\mathrm{T}}(b_t) \boldsymbol{R}_0(b_t) + \\ & b_t^2 \boldsymbol{R}_0^{\mathrm{T}}(b_t) (\boldsymbol{I} - b_t \boldsymbol{G}^{-1}\boldsymbol{D}) d_t + b_t^4 \boldsymbol{R}_0^{\mathrm{T}}(b_t) \boldsymbol{P}(b_t) d_t + \\ & b_t^4 \boldsymbol{R}_0^{\mathrm{T}}(b_t) \boldsymbol{R}_0(b_t) + b_t^3 E h^{\mathrm{T}} \boldsymbol{R}^{\mathrm{T}}(b_t) \boldsymbol{G}^{-1} h + b_t^4 E h^{\mathrm{T}} \boldsymbol{R}^{\mathrm{T}}(b_t) \boldsymbol{R}(b_t) h \end{aligned} \qquad (22)$$

Given the bound of $\boldsymbol{R}(b_t)$ and $\boldsymbol{R}_0(b_t)$, Eq. (22) can be simplified as

$$\begin{aligned} E(\|d_{t+1}\|^2 \mid D'^{(1)}, \cdots, D'^{(t)}) = {} & d_t^{\mathrm{T}} d_t - 2 b_t d_t^{\mathrm{T}} \boldsymbol{G}^{-1}\boldsymbol{D} d_t + b_t^2 d_t^{\mathrm{T}} (\boldsymbol{G}^{-1}\boldsymbol{D})^2 d_t + \\ & b_t^2 d_t^{\mathrm{T}} (\boldsymbol{P}(b_t) + \boldsymbol{P}^{\mathrm{T}}(b_t)) d_t + b_t^2 \boldsymbol{R}_0^{\mathrm{T}}(b_t) d_t + \\ & b_t^2 d_t^{\mathrm{T}} \boldsymbol{R}_0(b_t) + O(b_t^2) = d_t^{\mathrm{T}} d_t - 2 b_t d_t^{\mathrm{T}} \boldsymbol{G}^{-1}\boldsymbol{D} d_t + \\ & b_t^2 d_t^{\mathrm{T}} (\boldsymbol{G}^{-1}\boldsymbol{D})^2 d_t + b_t^2 d_t^{\mathrm{T}} \boldsymbol{Q}(b_t) d_t + \\ & b_t^2 \boldsymbol{R}_0^{\mathrm{T}}(b_t) d_t + b_t^2 d_t^{\mathrm{T}} \boldsymbol{R}_0(b_t) + O(b_t^2) \end{aligned} \qquad (23)$$

where $\boldsymbol{Q}(b_t) = \boldsymbol{P}(b_t) + \boldsymbol{P}^{\mathrm{T}}(b_t)$ is an $n\times n$ matrix and all their entries are bounded when b_t approaches zero. Furthermore, it can be seen that $\boldsymbol{Q}(b_t)$ is symmetric.

Let $l_{\min}$, $l_{\max}$ denote the minimal and maximal eigenvalues of $\boldsymbol{G}^{-1}\boldsymbol{D}$, respectively, and $l(b_t)$ denote the maximal eigenvalue of $\boldsymbol{Q}(b_t)$. Given that $\boldsymbol{G}^{-1}\boldsymbol{D}$ and $\boldsymbol{Q}(b_t)$ are symmetric, from the matrix theories[20], we have

$$
\begin{aligned}
&d_t^{\mathrm{T}}\boldsymbol{G}^{-1}\boldsymbol{D}d_t \ge l_{\min}\|d_t\|^2, \\
&d_t^{\mathrm{T}}(\boldsymbol{G}^{-1}\boldsymbol{D})^2 d_t = \|\boldsymbol{G}^{-1}\boldsymbol{D}d_t\|^2 \le \\
&\|\boldsymbol{G}^{-1}\boldsymbol{D}\|^2\|d_t\|^2 = l_{\max}^2\|d_t\|^2, \\
&d_t^{\mathrm{T}}\boldsymbol{Q}(b_t)d_t \le l(b_t)\|d_t\|^2
\end{aligned}
\tag{24}
$$

Therefore, from Formula (24), we get

$$
\begin{aligned}
&E(\|d_{t+1}\|^2 \mid D'^{(1)},\mathrm{L}\ \ ,D'^{(t)}) \le \\
&(1-2b_t l_{\min} + b_t^2 l_{\max}^2 + b_t^2 l(b_t))\|d_t\|^2 + \\
&b_t^2 \boldsymbol{R}_0^{\mathrm{T}}(b_t)d_t + b_t^2 d_t^{\mathrm{T}}\boldsymbol{R}_0(b_t) + O(b_t^2)
\end{aligned}
\tag{25}
$$

Taking expectations from both sides of Formula (25), following Lemma 3 we have

$$
\begin{aligned}
&E(\|d_{t+1}\|^2) \le (1-2b_t l_{\min} + b_t^2 l_{\max}^2 + \\
&b_t^2 l(b_t))E(\|d_t\|^2) + O(b_t^2)
\end{aligned}
\tag{26}
$$

Suppose the upper bound of $l(b_t)$ is l_0. From Formula (26) we have

$$
\begin{aligned}
&E(\|d_{t+1}\|^2) \le (1-2b_t l_{\min} + b_t^2 l_{\max}^2 + \\
&b_t^2 l_0)E(\|d_t\|^2) + O(b_t^2)
\end{aligned}
\tag{27}
$$

We can choose a positive real number u and a positive integer T such that for $\forall t \ge T$,

$$0 < u < 2l_{\min} - b_t(l_{\max}^2 + l_0) \tag{28}$$

and

$$b_t(2l_{\min} - b_t(l_{\max}^2 + l_0)) < 1 \tag{29}$$

Then Formula (27) implies that

$$E(\|d_{t+1}\|^2) \le (1-b_t u)E(\|d_t\|^2) + O(b_t^2) \tag{30}$$

From Lemma 4 and the assumption of b_t, we get

$$\lim_{t\to\infty}(E\|d_t\|^2) = 0 \tag{31}$$

The proof of Theorem 3 is completed.

In Ref. [10], Yao and Zhang has proved that CMAC learning algorithm will not converge if the learning rate is kept fixed in the case that training samples have noises, so the convergence of CA-CMAC will not be discussed in this work when the learning rate is kept fixed in this case.

3 Simulation

To testify the theorems derived above, a simulation is carried out to solve an inverse kinematics problem of 2-degree-of-freedom planar robot arm. Let (x,y) denote the coordinates of the gripper of the arm, and the joint angles are given by the following.

$$
\begin{cases}
q_1 = \arctan(y/x) + \arctan\ [l_2\sin q_2/(l_1 - l_2\cos q_2)], \\
q_2 = \arccos[(l_1^2 + l_2^2 - x^2 - y^2)/(2l_1 l_2)]
\end{cases}
\tag{32}
$$

where $\grave{e}_1$ and $\grave{e}_2$ are the first and second joint angles, 0 $\grave{e}_1,\grave{e}_2$ ð. Only $\grave{e}_1$ is learned. l_1 and l_2 are the link lengths. Suppose $l_1 = l_2 = 10$ and $0 < x,\ y < 10$. Simulation parameters are as follows: the intervals of x and y of the training sample are both 1, i.e., x, y = [0.5, 1.5, ⋯, 9.5]; the quantizing intervals of x and y are both 0.5; the generalization parameter is 8; the permitted error of each training sample is 0.001; the maximum cycle is 1000.

Simulation 1

In this simulation, the training samples are noiseless. Figure 1 shows the mean square error (MSE) of all training samples versus each cycle. It can be seen that when $0 < \hat{a} < 2.0$, the CA-CMAC learning scheme is convergent. But when $\hat{a} = 2.0$, the learning algorithm is not convergent. This confirms Theorem 2.

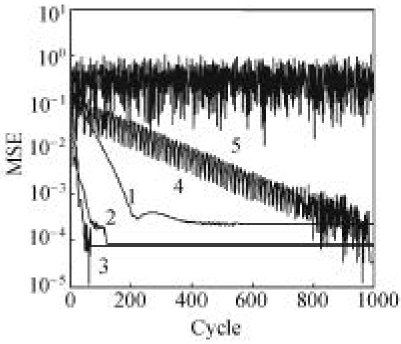

Fig. 1 The convergence property of CA-CMAC with different learning rates

1, $\hat{a}$ = 0.1; 2, $\hat{a}$ = 0.4; 3, $\hat{a}$ = 0.8; 4, $\hat{a}$ = 1.9; 5, $\hat{a}$ = 2.0

Simulation 2

In this simulation, when each sample is presented to CA-CMAC, a noise term is added to the desired output, which is a random variable generated by the Gaussian distribution N(0,0.05). The learning rate is adjusted dynamically as follows:

$$
\begin{cases}
b_t = b_{t-1}, & \text{if } \mathrm{MSE}_t < \mathrm{MSE}_{t-1}; \\
b_t = 0.8b_{t-1}, & \text{otherwise}
\end{cases}
\tag{33}
$$

where MSE_t means the MSE of all training samples after the t-th cycle. Figure 2 shows the MSE decreases and the learning algorithm converges when the learning rate is dynamically decreased. This confirms Theorem 3. Figure 2 also shows that the convergence speed of CA-CMAC is faster than that of CMAC while the convergence precisions of two algorithms are similar. If the required minimum MSE is 0.001, CA-CMAC achieves this goal after 55 cycles while CMAC needs about 150 cycles.

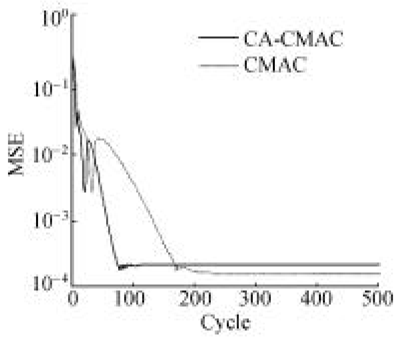

Fig. 2 The convergence of two algorithms when the training samples have noises

Simulation 3

In this simulation, the training samples are noiseless and the learning rate is dynamically decreased as Formula (33). The comparative study in Fig. 3 shows that the convergence property of CA-CMAC is better than that of CMAC both in convergence speed and in convergence precision. The fast convergence speed is especially important for many real-time control applications.

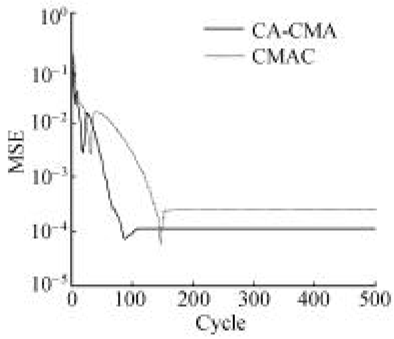

Fig. 3 The convergence of two algorithms when the training samples are noiseless

4 Conclusions

A credit-assignment CMAC algorithm is presented to reduce learning interference in machine learning. The convergence process of CA-CMAC is analyzed by presenting the credit matrix and the credit correlation matrix, and CMAC is proved to be a special case of CA-CMAC where all memory cells have the same credibility. The convergence theorems can be used as a guidance for choosing the learning rate properly in practical applications. The theorems also show that CA-CMAC has no additive convergence conditions compared with CMAC. The simulation results show that CA-CMAC has better convergence property than conventional CMAC, especially in the convergence speed. It is evident that CA-CMAC can be easily to be embedded in machines for self-learning and self-maintenance applications.

References

[1] Albus J S. A new approach to manipulator control: The cerebellar model articulation controller (CMAC). *Journal of Dynamic System, Measurement and Control*, 1975, **97**(3): 220-227.

[2] Albus J S. Data storage in the cerebellar model articulation controller (CMAC). *Journal of Dynamic System, Measurement and Control*, 1975, **97**(3): 228-233.

[3] Miller W T, Glanz F H, Kraft L G. CMAC: An associative neural network alternative to back propagation. *Proc. IEEE*, 1990, **78**:1561-1567.

[4] Linse D J, Stengel R F. Neural networks for function approximation in nonlinear control. *Amer. Control Conf.*, 1990, **1**: 674-679.

[5] Moody J. Fast learning in multi-resolution hierarchies. In: Touretzky D S, ed. Advance in Neural Information Processing Systems 1. Morgan Kaufmann, Los, Altos, CA, 1989: 29-39.

[6] Miller W T, Hewes P R, Glanz F H. Real-time dynamic control of an industrial manipulators using a neural network-based learning controllers. *IEEE Trans. Robot. Automat.*, 1990, **6**(1): 1-9.

[7] Cembrano G, Wells G, Sarda J. Dynamic control of a robot arm using CMAC neural networks. *Control Engineering Practice*, 1999, **5**: 485-492.

[8] Lin C S, Kim H. CMAC-based adaptive critic learning control. *IEEE Trans. Neural Network*, 1991, **2**: 530-535.

[9] Wong Y, Sideris A. Learning convergence in the cerebellar model articulation controller. *IEEE Trans. Neural Networks*, 1992, **3**: 115-121.

[10] Yao Shu, Zhang Bo. The learning convergence of CMAC in cyclic learning. *Journal of Computer Science* & *Technology*, 1994, **9**(4): 320-328.

[11] Lin C S, Chiang C T. Learning convergence of CMAC technique. *IEEE Trans. Neural Network,* 1997, **8**(6):

1281-1292.

[12] Lee J, Kramer B M. On the analysis of machine degradation using a neural network based pattern discrimination model. *SME Journal of Manufacturing Systems*, 1993, **12**(5): 379-387.

[13] Lee J. Teleservice engineering in manufacturing: Challenges and opportunities. *International Journal of Machine Tools & Manufacture*, 1998, **38**(8): 901-910.

[14] Lee J. Machine performance assessment methodology and advanced service technologies. In: 4th Frontiers in Engineering, National Academy Report, 1999: 75-83.

[15] Thompson D E. Neighborhood sequential and random training techniques for CAMC. *IEEE Trans-Neural Network*, 1995, **6**(1): 196-202.

[16] Sayil S, Lee K Y. A hybrid maximum error algorithm with neighborhood training for CMAC. In: Proceedings of the 2002 International Joint Conference on Neural Networks, 2002, **1**: 165-170.

[17] Su S F, Ted T, Hung T H. Credit assigned CMAC and its application to online learning robust controllers. *IEEE Trans. Systems, Man, and Cybernetics-Part B*: *Cybernetics*, 2003, **33**(2): 202-213.

[18] Xi Meicheng. Numerical Analysis Methods. Hefei, China: Chinese Science and Technology Publisher, 1995: 214-261. (in Chinese)

[19] Young D M. Iterative Solution of Large Linear Systems. New York: Academic Press, 1971.

[20] Shi Rongchang. Matrix Analysis. Beijing, China: Beijing Institute of Technology Publisher, 1996: 134-138. (in Chinese)

President of Kyoto University Visits Tsinghua University

Professor Kazuo Oike, President of Kyoto University, visited Tsinghua University on August 3, 2004. Tsinghua University President Gu Binglin met and had a friendly talk with professor Oike.

President Gu introduced the development of Tsinghua University to the guests and the two sides discussed the cooperation in the future.

After the meeting, Professor Oike paid visits to the Center for Advanced Studies and the Analysis Center.

Tsinghua University and Kyoto University have been in good relations for a long time. Now the two universities are taking the lead in a Sino-Japanese Core University Program on “urban environment”. Eight Japanese universities including University of Tokyo and six universities in China such as Peking University took part in the program.

Kyoto University was founded in 1897. It is the second university being established in Japan.

Reported by Li Han

(From http://news.cic.tsinghua.edu.cn)