Generalization ability of a perceptron with nonmonotonic transfer function

Jun-ichi Inoue,[1,3] Hidetoshi Nishimori,[1] and Yoshiyuki Kabashima[2]
[1]*Department of Physics, Tokyo Institute of Technology, Oh-okayama, Meguro-ku, Tokyo 152, Japan*
[2]*Department of Computational Intelligence and Systems Science, Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology, Yokohama 226, Japan*
[3]*Laboratory for Information Synthesis, Brain-Style Information Systems, RIKEN, Hirosawa 2-1, Wako-shi, Saitama 351-01, Japan*
(Received 20 May 1997; revised manuscript received 29 January 1998)

We investigate the generalization ability of a perceptron with nonmonotonic transfer function of a reversed-wedge type in on-line mode. This network is identical to a parity machine, a multilayer network. We consider several learning algorithms. By the perceptron algorithm the generalization error is shown to decrease by the $\alpha^{-1/3}$-law similarly to the case of a simple perceptron in a restricted range of the parameter a characterizing the nonmonotonic transfer function. For other values of a, the perceptron algorithm leads to the state where the weight vector of the student is just opposite to that of the teacher. The Hebbian learning algorithm has a similar property; it works only in a limited range of the parameter. The conventional AdaTron algorithm does not give a vanishing generalization error for any values of a. We thus introduce a modified AdaTron algorithm that yields a good performance for all values of a. We also investigate the effects of optimization of the learning rate as well as of the learning algorithm. Both methods give excellent learning curves proportional to α^{-1}. The latter optimization is related to the Bayes statistics and is shown to yield useful hints to extract maximum amount of information necessary to accelerate learning processes. [S1063-651X(98)01007-1]

PACS number(s): 87.10.+e

I. INTRODUCTION

In artificial neural networks, the issue of learning from examples has been one of the most attractive problems [1–4]. Traditionally emphasis has been put on the off-line (or batch) learning. In the off-line learning scenario, the student sees a set of examples (called a training set) repeatedly until equilibrium is reached. This learning scenario can be analyzed in the framework of equilibrium statistical mechanics based on the energy cost function, which means student's total error for a training set or on other types of cost functions [5–7]. However, recently, several important features of learning from examples were derived from the paradigm of on-line learning. In the on-line learning scenario, the student sees each example only once and throws it out, and he never sees it again. In other words, at each learning stage, the student receives a randomly drawn example and is not able to memorize it. The most recent example is used for modifying the student weight vector only by a small amount. The on-line learning has an advantage over the off-line counterpart in that it explicitly carries information about the current stage of achievement of the student as a function of the training time (which is proportional to the number of examples).

Over the past several years, many interesting results have been reported in relation to on-line learning. Among them, the generalization ability of multilayer networks is one of the central problems [8–10]. Multilayer neural networks are much more powerful machines for information representation than the simple perceptron.

Recently, the properties of neural networks with a nonmonotonic transfer function have also been investigated by several authors [11–16]. A perceptron with a nonmonotonic transfer function has the same input-output relations as a multilayer neural network called the parity machine. This parity machine has one hidden layer composed of three hidden units (the $K=3$ parity machine). The output of each unit is represented as $\mathrm{sgn}(-u)$, $\mathrm{sgn}(-a-u)$, and $\mathrm{sgn}(a-u)$, where $u \equiv \sqrt{N}(\mathbf{J}\cdot\mathbf{x})/|\mathbf{J}|$. Here $\mathbf{J}$ is the N-dimensional synaptic connection vector and $\mathbf{x}$ denotes the input signal. Then the final output of this machine is given as the product $\mathrm{sgn}(-u)\mathrm{sgn}(-a-u)\mathrm{sgn}(a-u)$. We regard this final output of the $K=3$ parity machine as the output of a perceptron with non monotonic transfer function. Recently, Engel and Reimers [17] investigated the generalization ability of this nonmonotonic perceptron following the off-line learning scenario. Their results are summarized as follows; For $0<a<\infty$, there exists a poor generalization phase with a large generalization error. As the number of presented patterns increases, a good generalization phase appears after a first order phase transition at some α. No studies have been made about the present system following the on-line learning scenario. In this paper we study the on-line learning process and the generalization ability of this nonmonotonic perceptron by various learning algorithms.

This paper is organized as follows. In the next section we introduce our model system and derive the dynamical equations with respect to two order parameters for a general learning algorithm. One is the overlap between the teacher and student weight vectors, and the other is the length of the student weight vector. In Sec. III, we investigate the dynamics of on-line learning in the nonmonotonic perceptron for the conventional perceptron learning and Hebbian leaning algorithms. We also investigate the asymptotic form of the differential equations in both small and large α limits and get the asymptotic behavior of the generalization error. In Sec. IV we investigate the AdaTron learning algorithm and modify the conventional AdaTron algorithm. In this modification procedure, we improve the weight function of the AdaTron learning so as to adopt it according to the range of a. In Sec. V, we optimize the learning rate and the general

weight function appearing in the on-line dynamics. As the weight function contains the variables unknown for the student, we average over these variables over distribution function unknown using the Bayes formula. Section VI contains concluding remarks.

II. THE MODEL SYSTEM AND DYNAMICAL EQUATIONS

We investigate the generalization ability of the nonmonotonic perceptrons for various learning algorithms. The student and teacher perceptron are characterized by their weight vectors, namely, $\mathbf{J}\in\Re^N$ and $\mathbf{B}\in\Re^N$ with $|\mathbf{B}|=1$, respectively. For a binary input signal $\mathbf{x}\in\{-1,+1\}^N$, the output is calculated by the nonmonotonic transfer function as follows:

$$T_a(v)=\mathrm{sgn}[v(a-v)(a+v)] \tag{1}$$

for the teacher and

$$S_a(u)=\mathrm{sgn}[u(a-u)(a+u)] \tag{2}$$

for the student, where we define the local fields of the teacher and student as $v\equiv\sqrt{N}(\mathbf{B}\cdot\mathbf{x})/|\mathbf{B}|$ and $u\equiv\sqrt{N}(\mathbf{J}\cdot\mathbf{x})/|\mathbf{J}|$, respectively. The on-line learning dynamics is defined by the following general rule for the change of the student vector under presentation of the mth example:

$$\mathbf{J}^{m+1}=\mathbf{J}^m+f(T_a(v),u)\mathbf{x}. \tag{3}$$

Well-known examples are the perceptron learning, $f=-S_a(u)\Theta(-T_a(v)S_a(u))$, the Hebbian learning, $f=T_a(v)$, and the AdaTron learning, $f=-ul\Theta(-T_a(v)S_a(u))$.

We rewrite the update rule, Eq. (3), of $\boldsymbol{J}$ as a set of differential equations introducing the dynamical order parameter describing the overlap between the teacher and student weight vectors $R^m\equiv(\mathbf{B}\cdot\mathbf{J}^m)/|\mathbf{J}^m|$ and another order parameter describing the norm of the student weight vector $l^m\equiv|\mathbf{J}^m|/\sqrt{N}$. By taking the overlap of both sides of Eq. (3) with $\mathbf{B}$ and by squaring both sides of the same equation, we obtain the dynamical equations in the limit of large m and N keeping $\alpha\equiv m/N$ finite as

$$\frac{dl}{d\alpha}=\frac{1}{2l}\langle\langle f^2(T_a(v),u)+2f(T_a(v),u)ul\rangle\rangle \tag{4}$$

and

$$\frac{dR}{d\alpha}=\frac{1}{l^2}\left\langle\left\langle -\frac{R}{2}f^2(T_a(v),u)-(Ru-v)f(T_a(v),u)l\right\rangle\right\rangle. \tag{5}$$

Here $\langle\langle\cdots\rangle\rangle$ denotes the average over the randomness of inputs

$$\langle\langle\cdots\rangle\rangle\equiv\int\int dudv(\cdots)P_R(u,v) \tag{6}$$

with

$$P_R(u,v)\equiv\frac{1}{2\pi\sqrt{1-R^2}}\exp\left(-\frac{(u^2+v^2-2Ruv)}{2(1-R^2)}\right). \tag{7}$$

As we are interested in the typical behavior under our training algorithm, we have averaged both sides of Eqs. (4) and (5) over all possible instances of examples. The Gaussian distribution (7) has been derived from the central limit theorem.

The generalization error, which is the probability of disagreement between the teacher and the trained student, is represented as $\epsilon_g=\langle\langle\Theta(-T_a(v)S_a(u))\rangle\rangle$. After simple calculations, we obtain the generalization error as

$$\begin{aligned}E(R)\equiv\epsilon_g=&2\int_a^\infty Dv H\left(\frac{a+Rv}{\sqrt{1-R^2}}\right)+2\int_a^\infty Dv H\left(\frac{-(a-Rv)}{\sqrt{1-R^2}}\right)+2\int_0^a Dv H\left(\frac{Rv}{\sqrt{1-R^2}}\right)-2\int_a^\infty Dv H\left(\frac{Rv}{\sqrt{1-R^2}}\right)\\&-2\int_0^a Dv H\left(\frac{a+Rv}{\sqrt{1-R^2}}\right)+2\int_0^a Dv H\left(\frac{a-Rv}{\sqrt{1-R^2}}\right),\end{aligned} \tag{8}$$

where we have set $H(x)=\int_x^\infty Dt$ with $Dt\equiv dt\exp(-t^2/2)/\sqrt{2\pi}$.

We would like to emphasize that the generalization error obtained in Eq. (8) is independent of the specific learning algorithm. In Fig. 1, we plot $E(R)=\epsilon_g$ for several values of a. This figure tells us that the student can acquire a perfect generalization ability if he is trained so that R converges to 1 for all values of a. We have confirmed also analytically that $E(R)$ is a monotonically decreasing function of R for any value of a.

III. HEBBIAN AND PERCEPTRON LEARNING ALGORITHMS

A. Hebbian learning

We first investigate the performance of the on-line Hebbian learning $f=T_a(v)$. We get the differential equations for l and R as follows:

$$\frac{dl}{d\alpha}=\left[\frac{1}{2}+\frac{2R}{\sqrt{2\pi}}(1-2\Delta)l\right]\Big/\ l, \tag{9}$$

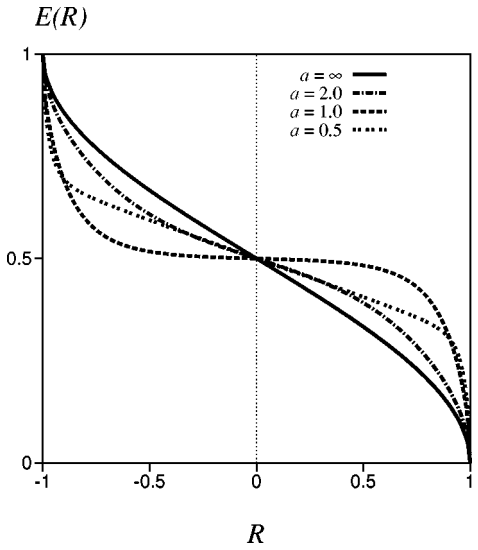

FIG. 1. Generalization error as a function of the overlap R for several values of a. The student should be trained so that the overlap goes to 1.

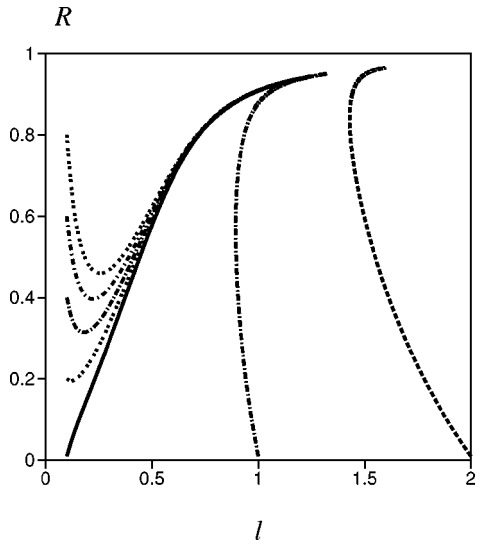

FIG. 2. Trajectories of the R-l flow for $a=\infty$. All R-l flows converge to the state of $R=1$ after an infinite number of examples are represented.

$$\frac{dR}{d\alpha}=\left[-\frac{R}{2}\frac{2}{\sqrt{2\pi}}(1-2\Delta)(1-R^2)l\right]\Big/\, l^2. \tag{10}$$

To determine whether or not R increases with α according to a, we approximate the differential equation for R around $R=0$ as

$$\frac{dR}{d\alpha}=\frac{2}{\sqrt{2\pi}}(1-2\Delta)\frac{1}{l^2}. \tag{11}$$

Therefore we use $R=1-\varepsilon$ for $a>a_c\equiv\sqrt{2\ln 2}$ and $R=\varepsilon-1$ for $a<a_c$. When $a>a_c$, we obtain

$$\epsilon_g=\frac{1}{\sqrt{2\pi}}\frac{1+2\Delta}{1-2\Delta}\frac{1}{\sqrt{\alpha}} \tag{12}$$

and

$$l=\sqrt{\frac{2}{\pi}}(1-2\Delta)\alpha. \tag{13}$$

On the other hand, for $a<a_c$ we obtain

$$\epsilon_g=1+\frac{1}{\sqrt{2\pi}}\frac{1+2\Delta}{1-2\Delta}\frac{1}{\sqrt{\alpha}} \tag{14}$$

and

$$l=-\sqrt{\frac{2}{\pi}}(1-2\Delta)\alpha. \tag{15}$$

We see that the Hebbian learning algorithms lead to the state $R=-1$ for $a<a_c$.

B. Perceptron learning

We next investigate the on-line perceptron learning $f=-S_a(u)\Theta(-T_a(v)S_a(u))$ by solving the next differential equations numerically;

$$\frac{dl}{d\alpha}=[\tfrac{1}{2}E(R)-F(R)l]/l, \tag{16}$$

$$\frac{dR}{d\alpha}=[-\tfrac{1}{2}E(R)R+(F(R)R-G(R))l]/l^2, \tag{17}$$

where $F(R)=\langle\langle\Theta(-T_a(v)S_a(u))S_a(u)u\rangle\rangle$ and $G(R)=\langle\langle\Theta(-T_a(v)S_a(u))S_a(u)v\rangle\rangle$. Using the distribution (7) we can rewrite these functions as

$$F(R)=\frac{(1-R)}{\sqrt{2\pi}}(1-2\Delta) \tag{18}$$

and

$$G(R)=-F(R), \tag{19}$$

where $\Delta\equiv\exp(-a^2/2)$. In Fig. 2 we plot the change of R and l as learning proceeds under various initial conditions for the case of $a=\infty$. We see that the student can reach the perfect generalization state $R=1$ for any initial condition. The R-l flow in the opposite limit $a=0$ is shown in Fig. 3. Apparently, for this case the student reaches the state with the weight vector opposite to the teacher, $R=-1$, after an infinite number of patterns are presented. From Eqs. (1) and (2), we should notice that the case of $a=0$ is essentially different from the case of a simple perceptron.

Since the two limiting cases, $a=\infty$ and $a=0$, follow different types of behavior, it is necessary to check what happens in the intermediate region. For this purpose, we first investigate the asymptotic behavior of the solution of Eqs. (16) and (17) near $R=\pm 1$ for large α. Using the notation $R=1-\varepsilon$, $\varepsilon\to 0$, the asymptotic forms of $E(R)$, $F(R)$, and $G(R)$ are found to be

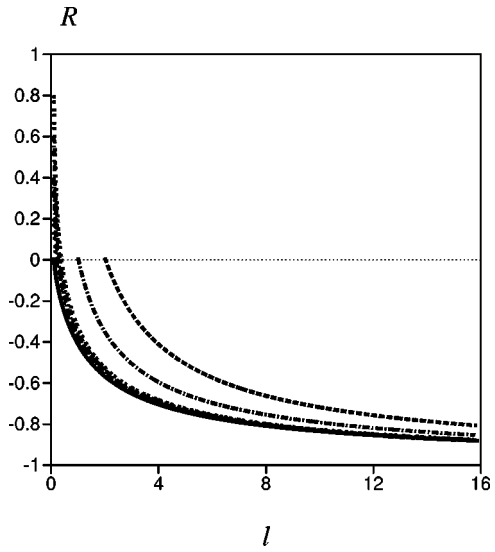

FIG. 3. Trajectories of the R-l flow for $a=0$. All R-l flows converge to the state $R=-1$. Therefore, the corresponding generalization error does not converge to the ideal value of zero for this case.

$$E(R)\simeq\frac{\sqrt{2\varepsilon}}{\pi}(1+2\Delta), \tag{20}$$

$$F(R)\simeq\frac{\varepsilon}{\sqrt{2\pi}}(1-2\Delta), \tag{21}$$

$$G(R)\simeq-\frac{\varepsilon}{\sqrt{2\pi}}(1-2\Delta). \tag{22}$$

Substituting these expressions into the differential equations (16) and (17), we obtain

$$\varepsilon=\left[\frac{(1+2\Delta)}{3\sqrt{2}(1-2\Delta)^2}\right]^{2/3}\alpha^{-2/3}, \tag{23}$$

$$l=\frac{1}{2\sqrt{\pi}}\left(\frac{1+2\Delta}{1-2\Delta}\right)\left[\frac{3\sqrt{2}(1-2\Delta)^2}{(1+2\Delta)}\right]^{1/3}\alpha^{1/3}. \tag{24}$$

Therefore, the generalization error is obtained from Eq. (20) as

$$\epsilon_g=(1+2\Delta)\frac{\sqrt{2}}{\pi}\left[\frac{(1+2\Delta)}{3\sqrt{2}(1-2\Delta)^2}\right]^{1/3}\alpha^{-1/3}. \tag{25}$$

The asymptotic form of l, Eq. (24), shows that Δ should satisfy $2\Delta<1$ or $a>a_c$. The assumption of $R=1-\varepsilon$ with $\varepsilon\to0$ thus fails if $a<a_c$. This fact can be verified from Eq. (17) expanded around $R=0$ as

$$\frac{dR}{d\alpha}\simeq\frac{2}{\sqrt{2\pi}}(1-2\Delta)\frac{1}{l^2}. \tag{26}$$

For $a<a_c$, R decreases with α. Therefore, we use the relation $R=\varepsilon-1,\varepsilon\to0$, instead of $R=1-\varepsilon$ for $a<a_c$. We then find the asymptotic form of the generalization error as

$$\epsilon_g=1+\left[\frac{1+2\Delta}{1-2\Delta}\right]\frac{1}{\sqrt{2\pi\alpha}} \tag{27}$$

and l goes to infinity as

$$l=-\frac{2}{\sqrt{2\pi}}(1-2\Delta)\alpha. \tag{28}$$

These two results, Eqs. (25) and (27), confirm the difference in the asymptotic behaviors between the two cases of $a=0$ and $a=\infty$.

We have found that the Hebbian and the conventional perceptron learning algorithms lead to the state $R=-1$ for $a<a_c=\sqrt{2\ln2}$. This antilearning effect may be understood as follows. If the student perceptron has learned only one example by the Hebb rule,

$$\mathbf{J}=T_{a=0}(v)\mathbf{x}. \tag{29}$$

Then the output of the student for the same example is

$$S_{a=0}(u)=-\mathrm{sgn}(u)=-\mathrm{sgn}(\mathbf{J}\cdot\mathbf{x})=-T_{a=0}(v). \tag{30}$$

This relation indicates the anti-learning effect for the $a=0$ case. Similar analysis holds for the perceptron learning.

C. Generalized perceptron learning

In this section, we introduce a multiplicative factor $|u|^\gamma$ in front of the perceptron learning function, $f=-|u|^\gamma\Theta(-T_a(v)S_a(u))S_a(u)$, and investigate how the generalization ability depends on the parameter γ. In particular, we are interested in whether or not an optimal value of γ exists. The learning dynamics is therefore

$$\mathbf{J}^{m+1}=\mathbf{J}^m-|u|^\gamma S_a(u)\Theta(-T_a(v)S_a(u))\mathbf{x}. \tag{31}$$

The case of $\gamma=0$ corresponds to the conventional perceptron learning algorithm. On the other hand, the case of $\gamma=1$ and $a\to\infty$ corresponds to the conventional AdaTron learning. Using the above learning dynamics, we obtain the differential equations with respect to l and R as

$$\frac{dl}{d\alpha}=\frac{1}{l}\left[\frac{E_G(R)}{2}-lF_G(R)\right], \tag{32}$$

$$\frac{dR}{d\alpha}=\frac{1}{l^2}\left[-\frac{R}{2}E_G(R)+[F_G(R)R-G_G(R)]l\right], \tag{33}$$

where $E_G(R)$, $F_G(R)$, and $G_G(R)$ are represented as

$$E_G(R)\equiv\langle\langle u^{2\gamma}\Theta(-T_a(v)S_a(u))\rangle\rangle, \tag{34}$$

$$F_G(R)\equiv\langle\langle|u|^{\gamma+1}\Theta(-T_a(v)S_a(u))S_a(u)\rangle\rangle \tag{35}$$

and

$$G_G(R)\equiv\langle\langle|u|^\gamma\Theta(-T_a(v)S_a(u))S_a(u)v\rangle\rangle. \tag{36}$$

Let us first investigate the behavior of the R-l flow near $R=0$. When R is very small, the right-hand side of Eq. (33) is found to be a γ-dependent constant:

$$\frac{dR}{d\alpha}=\frac{2^{1/2(\gamma-1)}}{\pi l}\Gamma\left(\frac{\gamma}{2}+\frac{1}{2}\right)(1-2\Delta), \tag{37}$$

where $\Gamma(x)$ is the gamma function. As the right hand side of Eq. (37) is positive for any γ as long as a satisfies $a>a_c$, R increases around $R=0$ only for this range of a. Thus the generalized perceptron learning algorithm succeeds in reaching the desired state $R=1$, not the opposite one $R=-1$, only for $a>a_c$, similarly to the conventional perceptron learning. Therefore, in this section we restrict our analysis to the case of $a>a_c$ and investigate how the learning curve changes according to the value of γ.

Using the notation $R=1-\varepsilon(\varepsilon\to 0)$, we obtain the asymptotic forms of E_G, F_G, and G_G as follows:

$$E_G\simeq c_1\varepsilon^{\gamma+1/2}+c_2\varepsilon^{1/2}, \tag{38}$$

$$F_G\simeq c_3\varepsilon^{1+\gamma/2}-c_4\varepsilon, \tag{39}$$

$$G_G\simeq -\frac{c_3}{\gamma+1}\varepsilon^{1+\gamma/2}+c_4\varepsilon, \tag{40}$$

where $c_1\equiv 2^{2\gamma+1/2}\Gamma(\gamma+1)/\pi(2\gamma+1)$, $c_2\equiv 4a^{2\gamma}\Delta/\sqrt{2}\pi$, $c_3\equiv 2^{\gamma+3/2}\Gamma(\gamma/2+3/2)/\pi(\gamma+2)$ and $c_4\equiv 2\Delta a^{\gamma}/\sqrt{2\pi}$. We first investigate the case of $\Delta\neq 0$ (finite a), namely, $c_2, c_4\neq 0$. The differential equations (32) and (33) are rewritten in terms of ε and $\delta=1/l$ as

$$\frac{d\delta}{d\alpha}=-\frac{\delta^3}{2}[c_1\varepsilon^{\gamma+1/2}+c_2\varepsilon^{1/2}]+\delta^2[c_3\varepsilon^{1+\gamma/2}-c_4\varepsilon], \tag{41}$$

$$\frac{d\varepsilon}{d\alpha}=\frac{\delta^2}{2}[c_1\varepsilon^{\gamma+1/2}+c_2\varepsilon^{1/2}]-\delta\left[\left(\frac{2+\gamma}{1+\gamma}\right)c_3\varepsilon^{1+\gamma/2}-2c_4\varepsilon\right]. \tag{42}$$

As $\gamma=0$ corresponds to the perceptron learning, we now assume $\gamma\neq 0$. When $\gamma>0$, the terms containing c_1 and c_3 can be neglected in the leading order. Dividing Eq. (41) by Eq. (42), we obtain

$$\frac{d\delta}{d\varepsilon}=\frac{\delta[-c_2\delta\varepsilon^{1/2}/2-c_4\varepsilon]}{[c_2\delta\varepsilon^{1/2}/2+2c_4\varepsilon]}. \tag{43}$$

If we assume $\delta\varepsilon^{1/2}\gg\varepsilon$ or $\delta\varepsilon^{1/2}\ll\varepsilon$, Eq. (43) is solved as $\delta=\exp(-\varepsilon)$, which is in contradiction to the assumption $|\delta|\ll 1$. Thus, we set

$$\delta=-\frac{4c_4}{c_2}\varepsilon^{1/2}+b\varepsilon^{c} \tag{44}$$

and determine b and $c(>1/2)$. Substituting Eq. (44) into (43), we find $b=8c_4/c_2(c_2,c_4>0)$ and $c=3/2$. The negative value of $\delta=1/l$ is not acceptable and we conclude that R does not approach 1 when $\gamma>0$.

Next we investigate the case of $\gamma<0$. Using the same technique as in the case of $\gamma>0$, we obtain

$$\varepsilon=\left[\frac{c_1(1+\gamma)(1-\gamma^2)}{6c_3^2(\gamma+2)}\right]^{2/3}\alpha^{-2/3}, \tag{45}$$

$$\delta=\frac{2c_3}{c_1}\left(\frac{\gamma+2}{\gamma+1}\right)\varepsilon^{(1/2)(1-\gamma)}-\frac{4c_3}{c_1(1-\gamma^2)}\varepsilon^{(1/2)(3-\gamma)}, \tag{46}$$

and

$$\epsilon_g=\frac{\sqrt{2}}{\pi}(1+2\Delta)\left[\frac{c_1(1+\gamma)(1-\gamma^2)}{6c_3^2(\gamma+2)}\right]^{1/3}\alpha^{-1/3}\equiv\frac{\sqrt{2}}{\pi}(1+2\Delta)f(\gamma)\alpha^{-1/3}. \tag{47}$$

We notice that γ should satisfy $-1<\gamma<0$, because the prefactor of the leading term of δ, namely, $(2c_3/c_1)(\gamma+2)/(\gamma+1)$, must be positive. As the prefactor of the generalization error increases monotonically from $\gamma=-1$ to $\gamma=0$, we obtain a smaller generalization error for γ closer to -1.

Next we investigate the case of $a\to\infty$, namely, $c_2, c_4=0$. We first assume $l\to l_0$ in the limit of $\alpha\to\infty$. In this solution, $dl/d\alpha=0$ should be satisfied asymptotically. Then, from Eq. (41), the two terms $\varepsilon^{\gamma+1/2}$ and $\varepsilon^{1+\gamma/2}$ should be equal to each other, namely, $\varepsilon^{\gamma+1/2}=\varepsilon^{1+\gamma/2}$, which leads to $\gamma=1$. The learning dynamics (31) with $a\to\infty$ and $\gamma=1$ is nothing but the AdaTron learning, which has already been investigated in detail [19]. The result for the generalization error is

$$\epsilon_g=\frac{3}{2\alpha} \tag{48}$$

if we choose l_0 as $l_0=1/2$, and

$$\epsilon_g=\frac{4}{3\alpha} \tag{49}$$

if we optimize l_0 to minimize the generalization error.

We next assume $l\to\infty$ as $\alpha\to\infty$. It is straightforward to see that ε has the same asymptotic form as in the case of $\Delta\neq 0$ and $\gamma<0$. Thus we have

$$\epsilon_g=\frac{\sqrt{2}}{\pi}f_2(\gamma)\alpha^{-1/3}, \tag{50}$$

where $f_2(\gamma)$ is defined as

$$f_2(\gamma)=\left[\frac{\pi(1+\gamma)(1-\gamma^2)\Gamma(\gamma+1)}{6(2^{5/2})\Gamma^2(\gamma/2+1/2)}\right]^{1/3} \tag{51}$$

and γ can take any value within $-1<\gamma<0$.

From the above analysis, we conclude that the student can get the generalization ability α^{-1} if and only if $a\to\infty$ and $\gamma=1$ (AdaTron). For other cases the generalization error behaves as $\alpha^{-1/3}$, the same functional form as in the case of the conventional perceptron learning, as long as the student can obtain a vanishing residual error. Therefore the learning

curve has universality in the sense that it does not depend on the detailed value of the parameter γ.

IV. ADATRON LEARNING ALGORITHM

A. AdaTron learning

In this subsection, we investigate the generalization performance of the conventional AdaTron learning $f=-ul\Theta(-T_a(v)S_a(u))$ [18]. The differential equations for l and R are given as follows:

$$\frac{dl}{d\alpha}=-\frac{l}{2}E_{\mathrm{Ad}}(R), \tag{52}$$

$$\frac{dR}{d\alpha}=\frac{R}{2}E_{\mathrm{Ad}}(R)-G_{\mathrm{Ad}}(R), \tag{53}$$

where $E_{\mathrm{Ad}}(R)=\langle\langle u^2\Theta(-T_a(v)S_a(u))\rangle\rangle$ and $G_{\mathrm{Ad}}(R)=\langle\langle uv\Theta(-T_a(v)S_a(u))\rangle\rangle$. After simple calculations, we obtain

$$E_{\mathrm{Ad}}(R)=2\left(\int_a^{\infty}+\int_{-a}^{0}\right)Duu^2H\left(\frac{a+Ru}{\sqrt{1-R^2}}\right)+2\left(\int_0^{a}+\int_{-\infty}^{-a}\right)Duu^2\left[H\left(\frac{Ru}{\sqrt{1-R^2}}\right)-H\left(\frac{a+Ru}{\sqrt{1-R^2}}\right)\right] \tag{54}$$

and

$$G_{\mathrm{Ad}}(R)=E_{\mathrm{Ad}}(R)R+\frac{4Ra\Delta}{\sqrt{2\pi}}(1-R^2)\left[H\left(\frac{a(1+R)}{\sqrt{1-R^2}}\right)-H\left(\frac{aR}{\sqrt{1-R^2}}\right)-H\left(\frac{a(1-R)}{\sqrt{1-R^2}}\right)+\frac{1}{2}\right]+\frac{2(1-R^2)^{3/2}}{\pi}$$
$$\times\left\{\Delta\exp\left[-\frac{a^2R^2}{2(1-R^2)}\right]-\Delta\exp\left[-\frac{a^2(1+R)^2}{2(1-R^2)}\right]-\Delta\exp\left[-\frac{a^2(1-R)^2}{2(1-R^2)}\right]+\exp\left[-\frac{a^2}{2(1-R^2)}\right]-\frac{1}{2}\right\}. \tag{55}$$

At first, we check the behavior of R around $R=0$. Evaluating the differential equation (53) around $R=0$, we obtain

$$\frac{dR}{d\alpha}=\frac{4}{\pi}\left(\Delta-\frac{1}{2}\right)^2. \tag{56}$$

From this result we find that for any value of a, the flow of R increases around $R=0$. In Fig. 4, we display the flows in the R-l plane for several values of a by numerical integration of Eq. (53). This figure indicates that the overlap R increases monotonically, but R does not reach the state $R=1$ if a is finite. This means that the differential equation (53) with respect to R has a nontrivial fixed point $R=R_0(<1)$ if $a<\infty$, which is the solution of the nonlinear equation $RE_{\mathrm{Ad}}(R)=2G_{\mathrm{Ad}}(R)$. Therefore, we conclude that for $a=\infty$ and $a=0$, we obtain the generalization error as $\epsilon_g\sim\alpha^{-1}$, but the generalization error converges to a finite value exponentially for finite a. In Fig. 5, we plot the corresponding gen-

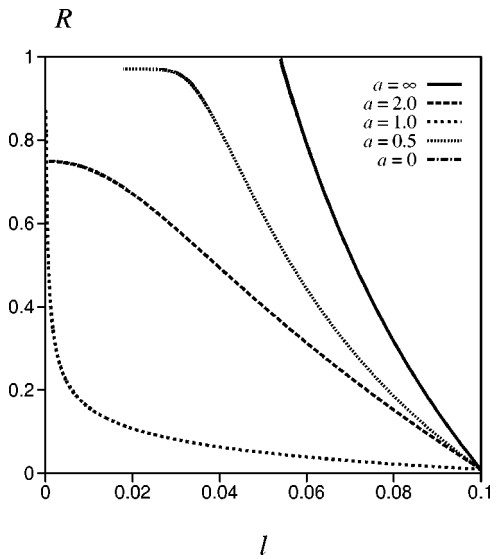

FIG. 4. Trajectories for the conventional AdaTron learning. Except for the case of $a=\infty$ and $a=0$ (overlapping), the trajectories converge to the state $l=0$.

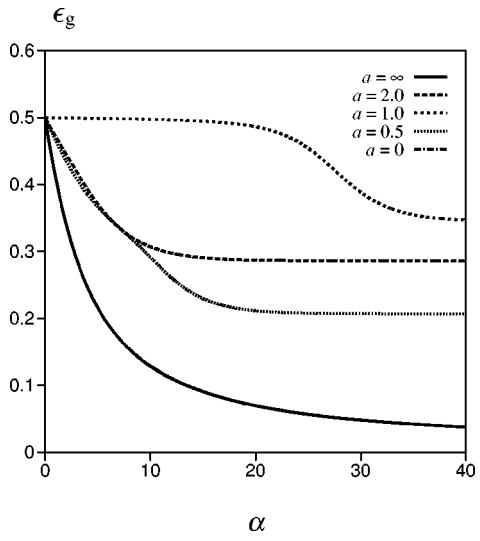

FIG. 5. Learning curves corresponding to Fig. 4. For the two cases of $a=\infty$ and $a=0$ (overlapping), the generalization errors converge to zero as α^{-1}. However, for the other cases, generalization errors converge to the finite value exponentially.

eralization error.

B. Modified AdaTron learning

In the previous subsection, we found that the on-line AdaTron learning fails to obtain the zero residual error for finite a. In this subsection, we modify the AdaTron learning as $f=\Theta(-T_a(v)S_a(u))h(u)l$ with

$$h(u)=\begin{cases} a-u & \left(u>\frac{a}{2}\right) \\ -u & \left(-\frac{a}{2}<u<\frac{a}{2}\right) \\ -a-u & \left(u<-\frac{a}{2}\right) \end{cases} \tag{57}$$

and see if the generalization ability of our nonmonotonic system is improved. The motivation for the above choice comes from the optimization of the learning algorithm to be mentioned in the next section. Details of derivation of Eq. (57) are found in the Appendix. Then the differential equation with respect to R is obtained as follows:

$$\frac{dR}{d\alpha}=-\frac{R^2}{2}E_{\mathrm{MA}}(R)-RF_{\mathrm{MA}}(R)+G_{\mathrm{MA}}(R), \tag{58}$$

where $E_{\mathrm{MA}}(R)=\langle\langle h^2(u)\Theta(-T_a(v)S_a(u))\rangle\rangle$, $F_{\mathrm{MA}}(R)=\langle\langle uh(u)\Theta(-T_a(v)S_a(u))\rangle\rangle$

and

$$G_{\mathrm{MA}}(R)=\langle\langle vh(u)\Theta(-T_a(v)S_a(u))\rangle\rangle.$$

To see the asymptotic behavior of the generalization error, we evaluate the leading-order contribution as R approaches 1, $R=1-\varepsilon$, as

$$E_{\mathrm{MA}}\sim\frac{2\sqrt{2}}{\pi}(1+2\Delta)\varepsilon^{3/2}, \tag{59}$$

$$F_{\mathrm{MA}}\sim-\frac{2\sqrt{2}}{\pi}(1+2(1-a^2)\Delta)\varepsilon^{3/2}, \tag{60}$$

$$G_{\mathrm{MA}}\sim\frac{4\sqrt{2}a^2\Delta}{\pi}\varepsilon^{3/2}. \tag{61}$$

Substituting these expressions into the differential equation (58), we obtain $\varepsilon^{1/2}=\sqrt{2}\pi/(1+2\Delta)\alpha^{-1}$ and the generalization error as

$$\epsilon_g=\frac{\sqrt{2}(1+2\Delta)}{\pi}\varepsilon^{1/2}=\frac{2}{\alpha}. \tag{62}$$

We should notice that the above result is independent of a and the generalization ability of the student is improved by this modification for all finite a.

V. OPTIMIZED LEARNING

A. Optimization of the learning rate

In the present subsection, we improve the conventional perceptron learning by introducing a time-dependent learning rate [20,19]. We consider the next on-line dynamics:

$$\mathbf{J}^{m+1}=\mathbf{J}^m-g(\alpha)\Theta(-T_a(v)S_a(u))S_a(u)\mathbf{x}. \tag{63}$$

Using the same technique as in the previous section, we can derive the differential equations with respect to l and R as follows:

$$\frac{dl}{d\alpha}=\frac{1}{l}\left[\frac{1}{2}g(\alpha)^2E(R)-g(\alpha)F(R)l\right], \tag{64}$$

$$\frac{dR}{d\alpha}=\frac{1}{l^2}\left[-\frac{R}{2}E(R)g(\alpha)^2+g(\alpha)(F(R)R-G(R))l\right]$$

$$\equiv L(g(\alpha)). \tag{65}$$

The optimal learning rate $g_{\mathrm{opt}}(\alpha)$ is determined so as to maximize $L(g(\alpha))$ to accelerate the increase of R. We then find

$$g_{\mathrm{opt}}=\frac{[F(R)R-G(R)]l}{RE(R)}. \tag{66}$$

Substituting this expression into the above differential equations, we obtain

$$\frac{dl}{dR}=-\frac{[F(R)R-G(R)][F(R)R+G(R)]l}{2R^2E(R)}, \tag{67}$$

$$\frac{dR}{dl}=\frac{[F(R)R-G(R)]^2}{2RE(R)}. \tag{68}$$

We can obtain the asymptotic form of $\varepsilon(=1-R)$, l, and ϵ_g with the same technique of analysis as in the previous section;

$$\varepsilon=4\left[\frac{2\sqrt{2}(1+2\Delta)}{(1-2\Delta)^2}\right]^2\alpha^{-2}, \tag{69}$$

$$l=\exp\left[-16\left(\frac{1+2\Delta}{(1-2\Delta)^2}\right)^4\alpha^{-4}\right], \tag{70}$$

and

$$\epsilon_g=\frac{\sqrt{2}}{\pi}(1+2\Delta)\left[\frac{2\sqrt{2}(1+2\Delta)}{(1-2\Delta)^2}\right]\alpha^{-1}. \tag{71}$$

Therefore, the generalization ability has been improved from $\alpha^{-1/3}$ for $g=1$ to α^{-1}. The optimal learning rate $g_{\mathrm{opt}}(\alpha)$ behaves asymptotically as

$$g_{\mathrm{opt}}=\frac{2\sqrt{2\pi}}{(1-2\Delta)}\alpha^{-1}\exp\left[-16\left(\frac{1+2\Delta}{(1-2\Delta)^2}\right)^4\alpha^{-4}\right]. \tag{72}$$

The factor $F(R)R-G(R)$ of $g_{\rm opt}$ appearing in Eq. (66) is calculated by substituting $F(R)$ and $G(R)$ in Eqs. (18) and (19) as $F(R)R-G(R)=(1-R^2)(1-2\Delta)/\sqrt{2\pi}$. Thus, at $a=a_c=\sqrt{2\ln 2}$, the optimal learning rate vanishes. Therefore our formulation does not work at $a=a_c$.

As the optimal learning rate $g_{\rm opt}$ changes the sign at $a=a_c$, from the arguments in Sec. III, we can see why the optimal learning rate can eliminate the antilearning.

In relation to this phenomenon at $a=\sqrt{2\ln 2}$, Van den Broeck [21,22] recently investigated the same reversed-wedge perceptron, which learns in the unsupervised mode from the distribution:

$$P(v)=2\frac{\exp(-v^2/2)}{\sqrt{2\pi}}[\Theta(v-a)+\Theta(v+a)\Theta(-v)] \tag{73}$$

with $v=\sqrt{N}(\mathbf{B}\cdot\mathbf{x})/|\mathbf{B}|$. For small α, he found $R(\alpha)\sim\sqrt{\alpha\langle v\rangle^2}$ for the optimal on-line learning, where $\langle\cdots\rangle$ denotes the average over the distribution (73). Then he showed that at $a=\sqrt{2\ln 2}$, the distribution (73) leads to $\langle v\rangle=0$ and consequently $R(\alpha)\equiv 0$. From this result, he concluded that as long as $\langle v\rangle=0$ holds, any kind of on-line learning necessarily fails and the corresponding learning curve has a plateau. It seems that a similar mechanism may lead to a failure of the optimal learning at $a=\sqrt{2\ln 2}$ in our model.

B. Optimization of the weight function using the Bayes formula

In this subsection we try another optimization procedure by Kinouchi and Caticha [23]. We choose the optimal weight function $f(T_a(v),u)$ by differentiating the right-hand side of Eq. (5) with the aim to accelerate the increase of R

$$f^*=\frac{l}{R}(v-Ru). \tag{74}$$

It is important to remember that f^* contains some unknown information for the student, namely, the local field of the teacher v. Therefore, we should average f^* over a suitable distribution to erase v from f^*. For this purpose, we transform the variables u and v to u and z:

$$v=z\sqrt{1-R^2}+Ru. \tag{75}$$

Then, the connected Gaussian distribution $P_R(u,v)$ is rewritten as

$$P_R(u,v)=\frac{1}{2\pi\sqrt{1-R^2}}\exp\left(-\frac{u^2}{2}\right)\exp\left(-\frac{z^2}{2}\right). \tag{76}$$

We then obtain

$$\langle f^*\rangle=\frac{\sqrt{1-R^2}}{R}l\langle z\rangle \tag{77}$$

where $\langle\cdots\rangle$ stands for the averaging over the variable v. Substituting this into the differential equation (5), we find

$$\frac{dR}{d\alpha}=\frac{(1-R^2)}{2R}\langle\langle z\rangle^2\rangle\rangle. \tag{78}$$

Let us now calculate $\langle z\rangle$. For this purpose, we use the distribution $P(z|y,u)$. This quantity means the posterior probability of z when y and u are given, where we have set $y\equiv T_a(v)$. This conditional probability is rewritten by the Bayes formula

$$P(z|y,u)=\frac{P(z)P(y|u,z)}{\int dz\,P(z)P(y|u,z)}, \tag{79}$$

from which we can calculate $\langle z\rangle$ as

$$\langle z\rangle=\int dz\,zP(z|y,u)=\frac{\int dz\,zP(z)P(y|u,z)}{\int dz\,P(z)P(y|u,z)}$$

$$=\frac{\int z\,Dz\,P(y|u,z)}{\int Dz\,P(y|u,z)}. \tag{80}$$

Here $P(y|u,z)$ is given as

$$P(y|u,z)=y\Theta(z\sqrt{1-R^2}+Ru)-y\Theta(z\sqrt{1-R^2}+Ru-a)$$
$$+y\Theta(-z\sqrt{1-R^2}-Ru-a)+\frac{1}{2}(1-y) \tag{81}$$

from the distribution $y=T_a(v)$. Then, the denominator of Eq. (79) is calculated as

$$\int Dz\,P(y|u,z)=y\int Dz\,\Theta(z\sqrt{1-R^2}+Ru)$$
$$-y\int Dz\,\Theta(z\sqrt{1-R^2}+Ru-a)$$
$$+y\int Dz\,\Theta(-z\sqrt{1-R^2}-Ru-a)$$
$$+\frac{1}{2}(1-y)$$
$$\equiv\Omega(y|u), \tag{82}$$

where $\Omega(y|u)$ means the posterior probability of y when the local field of the student u is given. As we treat the binary output teacher, we obtain from Eq. (82)

$$\Omega(\pm 1|u)=H\left(\mp\frac{Ru}{\sqrt{1-R^2}}\right)\mp H\left(\frac{a-Ru}{\sqrt{1-R^2}}\right)\pm H\left(\frac{a+Ru}{\sqrt{1-R^2}}\right). \tag{83}$$

In Figs. 6 ($R=0.5$) and 7, ($R=0.9$), we plot $\Omega(+1|u)$ for the cases of $a=4.0$, 2.0, 1.0 and $a=0.5$. From these figures, we find that for any $a\Omega(+1|u)$ seems to reach $[T_a(u)+1]/2$ as R goes to $+1$. Using the same technique, we can calculate $\int Dz\,zP(y|u,z)$ and obtain

$$\int Dz\,zP(y|u,z)=\frac{\sqrt{1-R^2}}{R}\frac{\partial}{\partial u}\Omega(y|u). \tag{84}$$

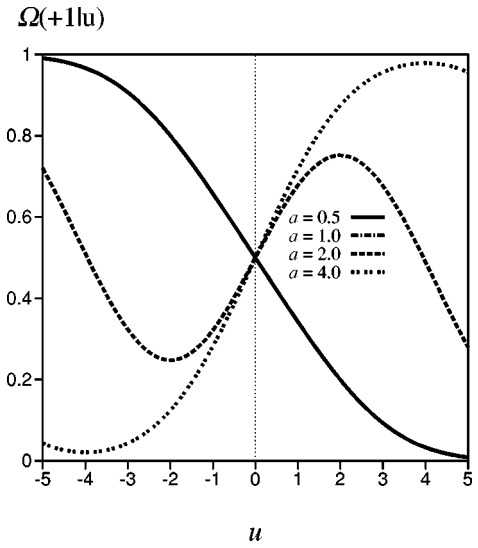

FIG. 6. Shapes of $\Omega(+1|u)$ for $R=0.5$.

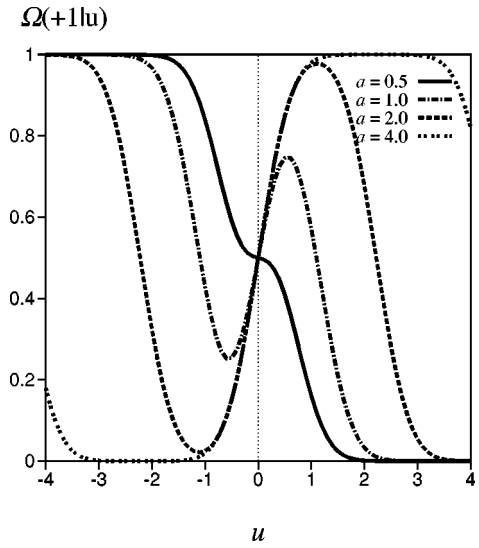

FIG. 7. Shapes of $\Omega(+1|u)$ for $R=0.8$. We see that for any a $\Omega(+1|u)$ seems to reach $[T_a(u)+1]/2$ as R goes to $+1$.

Substituting this into the right-hand side of $dR/d\alpha$, Eq. (78), we obtain

$$\frac{dR}{d\alpha}=\left\langle\left\langle -\frac{(1-R^2)^2}{2R^3}\left\{\frac{\partial}{\partial u}\ln\Omega(y|u)\right\}^2 + \frac{(1-R^2)^{3/2}z}{R^2}\frac{\partial}{\partial u}\ln\Omega(y|u)\right\rangle\right\rangle, \tag{85}$$

where $\langle\langle\cdots\rangle\rangle$ stands for the averaging over the distribution $P(y,u)=\int Dz\,P(y|u,z)P(u)P(z)$. Performing this average, we finally obtain

$$\frac{dR}{d\alpha}=\frac{(1-R^2)}{4\pi R}\int_{-\infty}^{\infty}Du\,\Xi_a(R,u), \tag{86}$$

where

$$\Xi_a(R,a)\equiv\left[\exp\left(-\frac{A_1^2}{2}\right)-\exp\left(-\frac{A_2^2}{2}\right)-\exp\left(-\frac{A_3^2}{2}\right)\right]^2\left[\frac{1}{H(-A_1)-H(A_2)+H(A_3)}+\frac{1}{H(A_1)+H(A_2)-H(A_3)}\right] \tag{87}$$

and $A_1\equiv Ru/\sqrt{1-R^2}$, $A_2\equiv(a-Ru)/\sqrt{1-R^2}$, $A_3\equiv(a+Ru)/\sqrt{1-R^2}$. We plot the generalization error by numerically solving Eqs. (16), (17), (67), (68), and (86) for the cases of $a=\infty$ in Fig. 8 and $a=1.0$ in Fig. 9. From these figures, we see that for both cases of $a=\infty$ and $a<\infty$, the generalization error calculated by the Bayes formula converges more quickly to zero than by the optimal learning rate $g_{\rm opt}(\alpha)$.

Recently, Simmonetti and Caticha [24] introduced the online learning algorithm for the nonoverlapping parity machine with general number of nodes K. In their method, the weight vector of the student in each hidden unit is trained by the method in Ref. [23]. In order to average over the internal fields of the teacher in the differential equation with respect to the specific hidden unit k of the student, they need the conditional probability that depends not only on the internal field of the unit k but also on the internal field of the other units $(i\neq k)$. This fact shows that their optimal algorithm is nonlocal. In our problem, the input-output relation of the machine can be mapped to those of a single layer reversed-wedge perceptron. Therefore, it is not necessary for us to use the information about all units and our optimizing procedure leads to a local algorithm.

In order to investigate the performance of the Bayes optimization, we have calculated the asymptotic form of the generalization error from Eq. (86) and the result is

$$\varepsilon^{1/2}=\frac{2}{(1+2\Delta)C\alpha} \tag{88}$$

for $\varepsilon=1-R$, where

$$C\equiv\frac{1}{\pi^{3/2}}\int_{-\infty}^{\infty}dt\,\frac{\exp(-t^2)}{H(t)}. \tag{89}$$

The generalization error is then given by Eq. (20) as

$$\epsilon_g=\frac{2\sqrt{2\pi}}{\displaystyle\int_{-\infty}^{\infty}dt\,\exp(-t^2)/H(t)}\,\frac{1}{\alpha}\sim 0.883\frac{1}{\alpha}. \tag{90}$$

This asymptotic form of the generalization error agrees with the result of Kinouchi and Caticha [23]. We notice that this form is independent of the width of the reversed wedge a.

We next mention the physical meaning of $\Xi_a(R,u)$ appearing in the differential equation (86). As the rate of in-

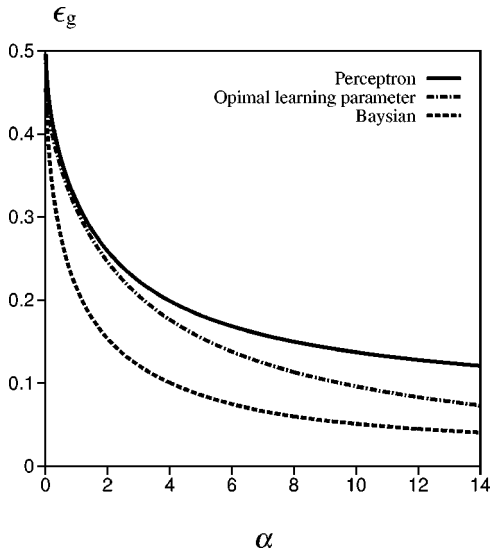

FIG. 8. Learning curves of perceptron, optimized perceptron, and Baysian optimization algorithms for $a=\infty$. The Baysian optimization algorithm is the best among the three.

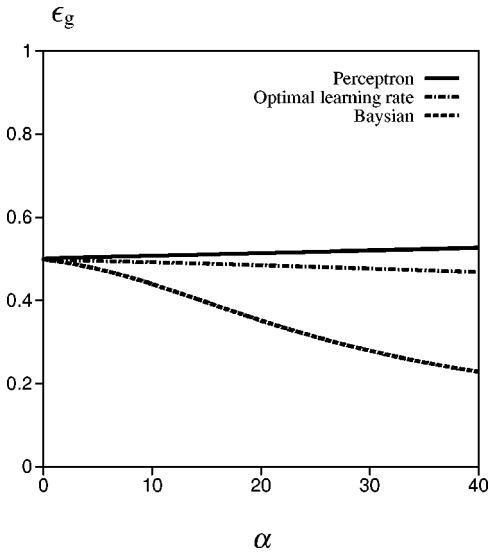

FIG. 9. Learning curves of perceptron, optimized perceptron, and Baysian optimization algorithms for $a=1.0$. The Baysian optimization algorithm gives the best result among the three.

crease $dR/d\alpha$ is proportional to $\Xi_a(R,u)$, this quantity is regarded as the distribution of the gain, which determines the increase of R. Therefore, $\Xi_a(R,u)$ yields important information about the strategy to make queries. A query means to restrict the input signal to the student, u, to some subspace. Kinzel and Ruján suggested that if the student learns by the Hebbian learning algorithm from restricted inputs, namely, inputs lying on the subspace $u=0$, the prefactor of the generalization error becomes half [25]. In the present formulation (86), query making can be incorporated by inserting appropriate delta functions in the integrand. The learning process is clearly accelerated by choosing the peak position of $\Xi_a(R,u)$ as the location of these delta functions. In Fig. 10 we plot the distribution $\Xi_a(R,u)$ for $a=2.0$ (top) and $a=0.8$ (bottom). From these figures, we learn that for large $a(=2.0)$, the most effective example lies on the decision boundary $(u=0)$ at the initial training stage (small R). However, as the student learns, two different peaks appear symmetrically and in the final stage of training, the distribution has three peaks around $u=0$ and $u=\pm a$. On the other hand, for small $a(=0.8)$, the most effective examples lie at the tails $(u=\pm\infty)$ for the initial stage. In the final stage, the distribution has two peaks around $u=\pm a$. Therefore it is desirable to change the location of queries adaptively.

VI. CONCLUSION

We have investigated the generalization abilities of a nonmonotonic perceptron, which may also be regarded as a multilayer neural network, a parity machine, in the on-line mode. We first showed that the conventional perceptron and Hebbian learning algorithms lead to the perfect learning $R=1$ only when $a>a_c=\sqrt{2\ln 2}$. The same algorithms yield the opposite state $R=-1$ in the other case $a<a_c$. These algorithms have originally been designed having the simple perceptron $(a=\infty)$ in mind, and thus are natural to give the opposite result for the reversed-output system $(a\sim 0)$. In contrast, the conventional AdaTron learning algorithm failed to obtain the zero residual error for all finite values of a. For the unlearnable situation (where the structures of the teacher and student are different), Inoue and Nishimori reported that the AdaTron learning converges to the largest residual error among the three algorithms [19]. It is interesting that the AdaTron learning algorithm is not useful even for the learnable situation.

In order to overcome this difficulty, we introduced several modified versions of the conventional learning rules. We first introduced the time-dependent learning rate into the on-line perceptron learning and optimize it. As a result, the generalization error converges to zero in proportion to α^{-1} except at $a=\sqrt{2\ln 2}$ where the learning rate becomes identically zero. We next improved the conventional AdaTron learning

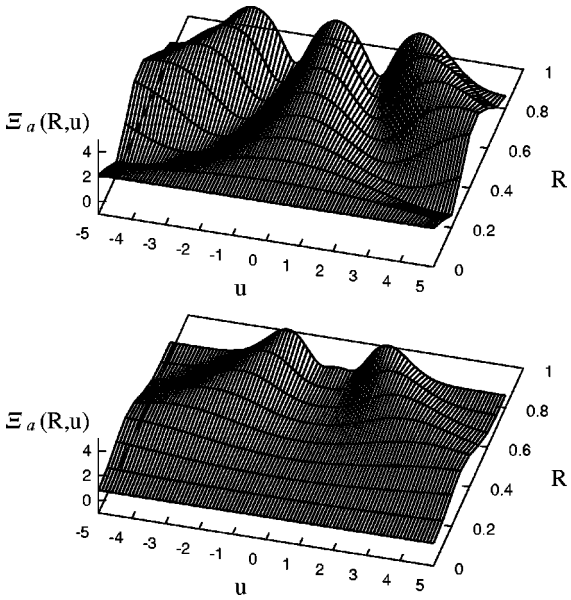

FIG. 10. Distributions of the gain $\Xi_a(R,u)$ for $a=2.0$ (top) and $a=0.8$ (bottom). The peak positions give the best place to make queries.

by modifying the weight function so that it changes according to the value of the internal potential u of the student. By this modification, the generalization ability of the student dramatically improved and the generalization error converges to zero with an a-independent form, $2\alpha^{-1}$.

We also investigated a different type of optimization: We first optimized the weight function $f(T_a(v),u)$ appearing in the on-line dynamics, not the rate g. Then, as the function f contains the unknown variable v, we averaged it over the distribution of v using the well-known technique of the Bayes statistics. This optimization procedure also provided other useful information for the student, namely, the distribution of most effective examples. Kinzel and Ruján [25] reported that for the situation in which a simple perceptron learns from a simple perceptron (the $a=\infty$ case), the Hebbian learning with selected examples ($u=0$) leads to faster convergence of the generalization error than the conventional Hebbian learning. However, we have found that for finite values of a, the most effective examples lie not only on the boundary $u=0$ but also on $u=\pm a$. Furthermore, we could learn that for small values of a and at the initial stage of learning (R small), the most effective examples lie on the tails ($u=\pm\infty$). As the learning proceeds, the most effective examples change the locations to $u=\pm a$. This information is useful for effective query constructions adaptively at each stage of learning.

ACKNOWLEDGMENTS

We thank Professor Shun-ichi Amari for useful discussions. One of the authors (J.I.) was partially supported by the Junior Research Associate Program of RIKEN. He also thanks Professor C. Van den Broeck for useful discussions. Y.K. was partially supported by the program ''Research for the future (RFTF)'' of Japan Society for the Promotion of Science.

APPENDIX: DERIVATION OF THE WEIGHT FUNCTION IN THE MODIFIED ADATRON LEARNING ALGORITHM

In this appendix, we explain how we introduced the modified weight function $\Theta(-T_a(v)S_a(u))h(u)l$ appearing in the AdaTron learning algorithm in Sec. IV B. From Eqs. (77) and (84) in Sec. V, the weight function using the Bayes formula is written as

$$\langle f^*\rangle = \frac{1-R^2}{R^2} l \frac{\partial}{\partial u} \ln \Omega(y|u). \tag{A1}$$

As this expression contains the unknown parameter R to the student, we try to find the suitable learning weight function, which agrees with the asymptotic form of $\langle f^*\rangle$ in the limit of $R\to 1$ [18]. For this purpose, we investigate the asymptotic form of $\Omega(y|u)$ as follows. We consider the cases of $T_a \equiv y=1$ and $y=-1$ separately.

(I) $y=1$. Using the relation $R=1-\varepsilon, \varepsilon\to 0$, we find

$$\begin{aligned}\Omega(y|u) &= H\left(-\frac{Ru}{\sqrt{1-R^2}}\right) - H\left(\frac{a-Ru}{\sqrt{1-R^2}}\right) + H\left(\frac{a+Ru}{\sqrt{1-R^2}}\right) \\ &\simeq \frac{1}{\sqrt{\pi}}\left[\operatorname{erfc}\left(\frac{-u}{2\sqrt{\varepsilon}}\right) - \operatorname{erfc}\left(\frac{a-u}{2\sqrt{\varepsilon}}\right) + \operatorname{erfc}\left(\frac{a+u}{2\sqrt{\varepsilon}}\right)\right].\end{aligned} \tag{A2}$$

The asymptotic form of $\Omega(y|u)$ depends on the range of u. For $u>a$, the asymptotic form of $\Omega(y|u)$ is

$$\Omega \sim \frac{1}{u-a}\sqrt{\frac{\varepsilon}{\pi}}\exp\left(-\frac{(u-a)^2}{4\varepsilon}\right). \tag{A3}$$

Therefore, $\langle f^*\rangle/l = -(u-a)$. Similarly, we find $\langle f^*\rangle/l = 0 (0<u<a$ and $u<-a)$, $\langle f^*\rangle/l = -u (-a/2<u<0)$, and $\langle f^*\rangle/l = -(u+a) (-a<u<-a/2)$.

(II) $y=-1$. Using the relation $R=1-\varepsilon$, we find for $u>a$

$$\Omega \sim 1 - \frac{1}{u-a}\sqrt{\frac{\varepsilon}{\pi}}\exp\left(-\frac{(u-a)^2}{4\varepsilon}\right). \tag{A4}$$

Therefore, the weight function $\langle f^*\rangle/l$ is 0 asymptotically. Similarly, we find $\langle f^*\rangle/l = 0 (a/2<u<a$ and $-a<u<0)$, $\langle f^*\rangle/l = -u (0<u<a/2)$, and $\langle f^*\rangle/l = -(a+u)(u<-a)$.

From the results of (I) and (II), we find the modified AdaTron learning algorithm as

$$\mathbf{J}^{m+1} = \mathbf{J}^m + \Theta(-T_a(v)S_a(u))h(u)l\mathbf{x}, \tag{A5}$$

where

$$h(u) = \begin{cases} a-u & \left(u>\dfrac{a}{2}\right) \\ -u & \left(-\dfrac{a}{2}<u<\dfrac{a}{2}\right) \\ -a-u & \left(u<-\dfrac{a}{2}\right). \end{cases} \tag{A6}$$

[1] S. Amari, IEEE Trans. Electromagn. Compat. **16**, 299 (1967).

[2] J. A. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation* (Addison-Wesley, Redwood City, 1991).

[3] T. H. L. Watkin, A. Rau, and M. Biehl, Rev. Mod. Phys. **65**, 499 (1993).

[4] M. Opper and W. Kinzel, in *Physics of Neural Networks III*, edited by E. Domany, J. L. van Hemmen, and K. Schulten (Springer, Berlin, 1995).

[5] M. Griniasti and H. Gutfreund, J. Phys. A **24**, 715 (1991).

[6] R. Meir and J. F. Fontanari, Phys. Rev. A **45**, 8874 (1992).

[7] O. Kinouchi and N. Caticha, Phys. Rev. E **54**, R54 (1996).

[8] Y. Kabashima, J. Phys. A **27**, 1917 (1994).

[9] H. Sompolinsky, N. Barkai, and H. S. Seung, *Neural Net-*

works: The Statistical Mechanics Perspective, edited by J. H. Oh, C. Kwon, and S. Cho (World Scientific, Singapore, 1995).
[10] D. Saad and S. A. Solla, Phys. Rev. E **52**, 4225 (1995).
[11] M. Morita, S. Yoshizawa, and K. Nakano, Trans. Inst. Electron. Inf. Commun. Eng. C-I **J73-D-II**, 242 (1993) (in Japanese).
[12] H. Nishimori and I. Opris, Neural Networks **6**, 1061 (1993).
[13] J. Inoue, J. Phys. A **29**, 4815 (1996).
[14] M. Morita, Neural Networks **9**, 1477 (1996).
[15] G. Boffetta, R. Monasson, and R. Zecchina, J. Phys. A **26**, L507 (1993).
[16] R. Monasson and D. O'Kane, Europhys. Lett. **27**, 85 (1994).
[17] A. Engel and L. Reimers, Europhys. Lett. **28**, 531 (1994).
[18] M. Biehl and P. Riegler, Europhys. Lett. **28**, 525 (1994).
[19] J. Inoue and H. Nishimori, Phys. Rev. E **55**, 4544 (1997).
[20] J. Inoue, H. Nishimori, and Y. Kabashima, J. Phys. A **30**, 3795 (1997).
[21] C. Van den Broeck, *Proc. of Theoretical Aspects of Neural Computation 97 (TANC-97)* (Springer-Verlag, Berlin, in press).
[22] C. Van den Broeck and P. Reimann, Phys. Rev. Lett. **76**, 2188 (1996).
[23] O. Kinouchi and N. Caticha, J. Phys. A **26**, 6243 (1992).
[24] R. Simonetti and N. Caticha, J. Phys. A **29**, 4859 (1996).
[25] W. Kinzel and P. Ruján, Europhys. Lett. **13**, 473 (1990).