On-line Gibbs learning. II. Application to perceptron and multilayer networks

J. W. Kim and H. Sompolinsky
Racah Institute of Physics and Center for Neural Computation, Hebrew University, Jerusalem 91904, Israel
(Received 23 December 1997)

In the preceding paper (''On-line Gibbs Learning. I. General Theory'') we have presented the on-line Gibbs algorithm (OLGA) and studied analytically its asymptotic convergence. In this paper we apply OLGA to on-line supervised learning in several network architectures: a single-layer perceptron, two-layer committee machine, and a winner-takes-all (WTA) classifier. The behavior of OLGA for a single-layer perceptron is studied both analytically and numerically for a variety of rules: a realizable perceptron rule, a perceptron rule corrupted by output and input noise, and a rule generated by a committee machine. The two-layer committee machine is studied numerically for the cases of learning a realizable rule as well as a rule that is corrupted by output noise. The WTA network is studied numerically for the case of a realizable rule. The asymptotic results reported in this paper agree with the predictions of the general theory of OLGA presented in paper I. In all the studied cases, OLGA converges to a set of weights that minimizes the generalization error. When the learning rate is chosen as a power law with an optimal power, OLGA converges with a power law that is the same as that of batch learning. [S1063-651X(98)07908-2]

PACS number(s): 87.10.+e

I. INTRODUCTION

Most of the conventional on-line learning algorithms are variations of the stochastic gradient descent algorithm which moves in the direction of the gradient descent of the instantaneous error function [1–17]. For a sufficiently small learning rate, the stochastic gradient descent algorithm converges to a local minimum of the generalization error. As this algorithm relies on differentiating the error function, it is inapplicable to learning Boolean functions or other discrete valued functions which are extremely useful for decision and classification tasks. In a previous paper (paper I), we have presented a model of on-line learning, which we called the on-line Gibbs algorithm (OLGA). This model is also applicable to learning discrete valued functions. In paper I we have analyzed its general asymptotic properties and showed that this algorithm converges in the limit of infinite number of examples to a local minimum of the generalization error for both realizable and unrealizable tasks. Furthermore, with an appropriate choice of a power-law learning rate its asymptotic convergence obeys, in general, similar power laws as those obtained in batch learning.

Computing the one-step update of the weights according to OLGA may be rather complex, depending on the system architecture. It is therefore important to know whether OLGA can be readily implemented in network architectures which are commonly used in supervised learning. In this paper we study the application of OLGA to several network architectures. Our first goal is to derive explicit update rules for these architectures. The second goal is to study the convergence properties of the algorithm in these systems for various target rules and to demonstrate some of the general results derived in paper I. The general definition of OLGA includes a temperature parameter characterizing its stochastic nature. We will focus in this paper only on the deterministic version of OLGA, i.e., its zero-temperature limit.

The outline of this paper is as follows. In the following section, we briefly summarize the definition of OLGA. In Sec. III we apply OLGA to a single-layer perceptron and study analytically and numerically its behavior for a realizable target rule as well as various kinds of unrealizable rules. The analytical studies are based on mean-field theory which is valid in the thermodynamic (large network size) limit. In Sec. IV we define the OLGA update rule in the case of a two-layer network with a committee-machine architecture. We study numerically this algorithm for learning a realizable rule and a rule corrupted by output noise. In Sec. V we apply OLGA to the winner-takes-all (WTA) classifier. We study numerically the case of a WTA network learning examples generated by a network with the same architecture. In Sec. VI we summarize and discuss the results.

II. DEFINITION OF OLGA

We consider a learning system defined by a function $\sigma(\mathbf{s};\mathbf{w})$, where $\mathbf{s}$ is the input vector and σ is the output, which for simplicity is taken as a scalar. The target task is a real valued function $\sigma_0(\mathbf{s})$. At each presentation of an example, indexed by the integer n, the system is given an input vector $\mathbf{s}^n$ and its desired output $\sigma_0^n=\sigma_0(\mathbf{s}^n)$. The inputs are drawn at random from a distribution $D\mathbf{s}$. For each example there is an error function $\epsilon(\mathbf{w};\mathbf{s})$ which measures the dissimilarity between the system output σ and the desired value σ_0. We denote by $\mathbf{w}$ the current weight vector, i.e., the weights evaluated after $n-1$ presentations of examples, and by $\mathbf{w}'$ the new weight vector, which is evaluated following the presentation of the nth example $\mathbf{s}=\mathbf{s}^n$. Given $\mathbf{w}$ and $\mathbf{s}$ the update rule for evaluating $\mathbf{w}'$ is based on the energy function

$$E(\mathbf{w}'|\mathbf{w},\mathbf{s})=\epsilon(\mathbf{w}';\mathbf{s})+\frac{\lambda}{2}\|\mathbf{w}'-\mathbf{w}\|^2. \tag{1}$$

It is an energy function in the space of the new weights $\mathbf{w}'$ which depends parametrically on $\mathbf{w}$ and $\mathbf{s}$. The first term in E represents the cost incurred by the error due to the new example. Minimizing this instantaneous error is not a good

strategy as it will lead to large changes in the values of the weights which will quickly erase past knowledge stored in the current weights. In order to avoid such changes we add the last term in E which prevents the system from making big changes in the weights at each presentation. Thus E represents a compromise between the need to satisfy the new example and the need to minimize the changes in the weights at each step.

In general OLGA is defined as a stochastic learning rule. Given the current weights and the new randomly sampled example using $D\mathbf{s}$, the new weight vector is sampled at random with the (conditional) on-line Gibbs distribution, i.e.,

$$\mathcal{P}(\mathbf{w}'|\mathbf{w},\mathbf{s}) \propto \exp\left(-\frac{\beta}{2}E(\mathbf{w}'|\mathbf{w},\mathbf{s})\right). \tag{2}$$

In this paper we will focus on the deterministic limit of OLGA which corresponds to the case $T=1/\beta=0$. We will consider here only cases *where the instantaneous measure of error is binary*. In this case, minimizing E implies that the current weight $\mathbf{w}$ is changed only if $\mathbf{w}$ does not satisfy the new example and in addition there is a weight vector sufficiently close to $\mathbf{w}$ that does satisfy the new example. Specifically, the $T=0$ OLGA for the binary error consists of three principles.

(1) Error correction. If $\epsilon(\mathbf{w};\mathbf{s})=0$ then the minimum of E is clearly $\mathbf{w}'=\mathbf{w}$, hence no change is made. Furthermore when $\epsilon(\mathbf{w};\mathbf{s})=1$, and a move is made, it will always be to a new weight vector that does satisfy the new example. Whether such a move is performed depends on the two following rules.

(2) Minimal change. If $\epsilon(\mathbf{w};\mathbf{s})=1$ then one has to search for the *nearest* vector to $\mathbf{w}$ that *satisfies* the new example.

(3) Bounded change. This new vector is chosen as $\mathbf{w}'$ provided that it lies within a hypersphere centered on $\mathbf{w}$ with radius $\sqrt{2/\lambda}$, i.e.,

$$\|\mathbf{w}'-\mathbf{w}\| < \sqrt{2/\lambda}. \tag{3}$$

Otherwise, $\mathbf{w}'=\mathbf{w}$.

III. OLGA FOR A SINGLE-LAYER PERCEPTRON

In this section we apply OLGA to the case where the learning system is a single-layer perceptron [18],

$$\sigma(\mathbf{w};\mathbf{s}) = \mathrm{sgn}(\mathbf{w}\cdot\mathbf{s}), \tag{4}$$

where both $\mathbf{s}$ and $\mathbf{w}$ are N-component vectors. It is assumed that the learned rule is also a dichotomy, i.e., $\sigma_0=\pm 1$. We first define the learning algorithm. We then present analytical and numerical results for the learning curve of this algorithm in specific cases of realizable and unrealizable target rules.

A. Definition of the algorithm

We assume that at the nth step, the perceptron is given a new example in the form of an input vector $\mathbf{s}^n$ and a label σ_0^n. Since the output is independent of the *magnitude* of the weight vector $\mathbf{w}$, we will use a version of OLGA that normalizes the weight vector at each step. We will show below that the normalized OLGA reduces to the following update rule:

$$\mathbf{w}^n = \begin{cases} A_n(\mathbf{w}^{n-1} - h_{n-1}\hat{\mathbf{s}}^n), & 0 < -h_{n-1}\sigma_0^n < \sqrt{2/\lambda[1-1/(2\lambda N)]} \\ \mathbf{w}^{n-1} & \text{otherwise,} \end{cases} \tag{5}$$

where the quantity h_{n-1} is the local field of the current weight $\mathbf{w}^{n-1}$ induced by $\mathbf{s}^n$,

$$h_{n-1} \equiv \mathbf{w}^{n-1}\cdot\hat{\mathbf{s}}^n, \tag{6}$$

where $\hat{\mathbf{s}}\equiv\mathbf{s}/\sqrt{\mathbf{s}\cdot\mathbf{s}}$, and the normalization coefficient A_n is

$$A_n = (1-N^{-1}h_{n-1}^2)^{-1/2}. \tag{7}$$

The upper bound on $|h_{n-1}|$ in Eq. (5) holds for $\lambda > 1/N$. If $\lambda < 1/N$ there is no upper bound on $|h_{n-1}|$.

We now derive the above update rule. In order to minimize the energy E, Eq. (1), we measure the local field h_{n-1} of the current weight $\mathbf{w}^{n-1}$ induced by $\mathbf{s}^n$. If $\sigma_0 h_{n-1} > 0$, i.e., $\sigma=\sigma_0$, we keep the current weight $\mathbf{w}^{n-1}$. If $\sigma\neq\sigma_0$, i.e., $\sigma_0 h_{n-1} < 0$, we search a *minimal* new weight vector $\mathbf{w}^n$ which satisfies $\sigma_0 h_n > 0$. Since changing the components of $\mathbf{w}^{n-1}$ which are orthogonal to $\mathbf{s}^n$ will increase the second term in E without contributing to the correction of the instantaneous error, the principle of minimal change implies that these components are unchanged, i.e.,

$$\mathbf{w}^n = \mathbf{w}^{n-1} + (h_n - h_{n-1})\hat{\mathbf{s}}^n. \tag{8}$$

In order to minimize the change in $\mathbf{w}$ we have to make h_n arbitrarily close to zero with a sign such that $\sigma_0 h_n \geqslant 0$. Substituting $h_n=0$ in Eq. (8) yields

$$\mathbf{w}^n = \mathbf{w}^{n-1} - h_{n-1}\hat{\mathbf{s}}^n. \tag{9}$$

Incorporating normalization of weight vectors the minimal weight vector is given by

$$\mathbf{w}^n = A_n(\mathbf{w}^{n-1} - h_{n-1}\hat{\mathbf{s}}^n). \tag{10}$$

The value of A_n is determined by requiring that $\mathbf{w}^n\cdot\mathbf{w}^n = N$, yielding Eq. (7) . Finally, the bounded change condition, Eq. (3), reads $2/\lambda > \|\Delta\mathbf{w}\|^2 = 2N(1-1/A_n)$. Incorporating this requirement, the updating rule, Eq. (10) is implemented if and only if

$$0 < -h_{n-1}\sigma_0^n < \sqrt{2/\lambda[1-1/(2\lambda N)]} \tag{11}$$

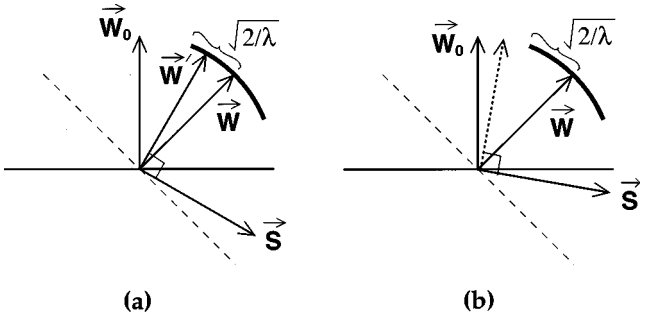

FIG. 1. Schematic illustration of OLGA update rule for a single-layer perceptron. The vectors $\mathbf{w}_0$ and $\mathbf{w}$ are the teacher and the student weights, respectively. The dashed line denotes the decision boundary of the student. The vector $\mathbf{s}$ is the new input, and we show the case of $\sigma_0(\mathbf{s})=-1$; $\sigma(\mathbf{w},\mathbf{s})=1$. The vector $\mathbf{w}'$ is the nearest vector which satisfies $\sigma_0(\mathbf{s})$. (a) If $\mathbf{w}'$ lies within a distance of $\sqrt{2/\lambda}$ from $\mathbf{w}$, $\mathbf{w}$ moves to $\mathbf{w}'$. (b) Otherwise, no update is made.

for $\lambda > 1/N$. If $\lambda < 1/N$ there is no upper bound on $|h_{n-1}|$. If these conditions (11) are not satisfied, $\mathbf{w}^n = \mathbf{w}^{n-1}$. Finally, we note that for large λN the bound simplifies to

$$0 < -h_{n-1}\sigma_0^n < \sqrt{2/\lambda}. \tag{12}$$

The perceptron update rule is shown schematically in Fig. 1.

B. Learning a perceptron rule

We consider the problem of a single-layer perceptron learning a realizable rule. The labeled examples are generated by a ''teacher'' perceptron with a weight vector $\mathbf{w}_0$. The minimum value of ϵ_g, $\epsilon_{\min}$, is zero if $\mathbf{w}=\mathbf{w}_0$. From the general theoretical results of paper I we expect that OLGA will converge to the teacher vector for all values of λ, with a generalization error that vanishes as $1/n$. In the following we check this prediction by solving analytically the dynamics of OLGA in the limit of large N and by numerical simulations. These calculations are performed for Gaussian input distribution with zero mean and unit variance $\langle\langle s_i^2\rangle\rangle = 1$. For this input distribution, the generalization error ϵ_g is

$$\epsilon_g(\mathbf{w}) = \frac{1}{\pi}\arccos(R), \tag{13}$$

where the overlap R between $\mathbf{w}$ and $\mathbf{w}_0$ is

$$R = \frac{1}{N}\mathbf{w}\cdot\mathbf{w}_0. \tag{14}$$

We can derive the expression for $\Delta R_n \equiv R_n - R_{n-1}$ in the large N limit using Eqs. (5) and (7). Expanding A_n in powers of $1/N$ yields

$$\mathbf{w}^n \approx \left(1+\frac{1}{2N}h_{n-1}^2\right)\mathbf{w}^{n-1} - h_{n-1}\hat{\mathbf{s}}^n. \tag{15}$$

Taking the inner product with $\mathbf{w}_0$ on both sides of the above equation yields

$$\Delta R_n = \frac{1}{N}\left(\frac{1}{2}R_{n-1}\langle h_{n-1}^2\rangle_D - \langle h_{n-1}^0 h_{n-1}\rangle_D\right), \tag{16}$$

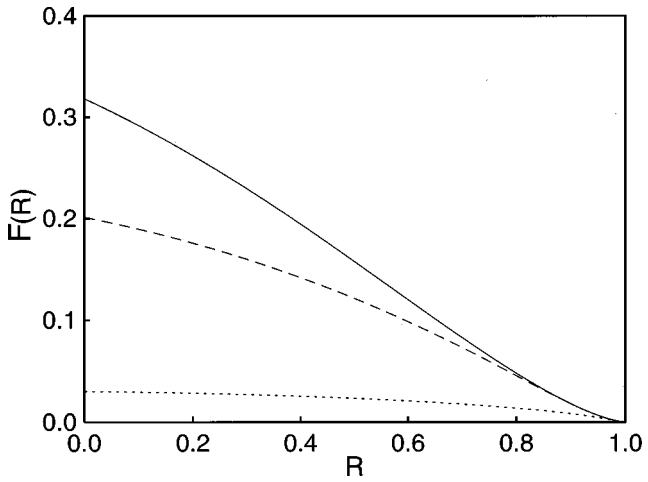

FIG. 2. $F(R)$, Eq. (18), for $\lambda=0.1$ (solid line), 1.0 (dashed line), and 10.0 (dotted line).

where $h_{n-1}^0 = \mathbf{w}_0\cdot\hat{\mathbf{s}}^n$ and D denotes the region in the space of $\mathbf{s}^n$ which satisfies the condition given by Eq. (12). The average $\langle\ \rangle_D$ can be explicitly performed using the Gaussian statistics of h and h^0, yielding

$$\Delta R_n = \frac{1}{N}F(R_{n-1}), \tag{17}$$

where

$$F(R) \equiv -R\int_0^{\sqrt{2/\lambda}} Dy\, y^2 H\left(\frac{Ry}{\sqrt{1-R^2}}\right) + \frac{1}{\pi}(1-R^2)^{3/2}\left[1-\exp\left(-\frac{1}{\lambda(1-R^2)}\right)\right], \tag{18}$$

where y represents the random variable h_{n-1} and

$$H(x) = \int_x^\infty Dy \tag{19}$$

and $Dy = dy\exp(-y^2/2)/\sqrt{2\pi}$. Finally, in the large N limit we can define a continuous scaled time variable

$$\alpha = \frac{n}{N} \tag{20}$$

and write $\Delta R_n = dR/d\alpha$, thereby obtaining the following differential equation for $R(\alpha)$:

$$\frac{dR}{d\alpha} = F(R), \tag{21}$$

where $F(R)$ is defined as Eq. (18).

The shape of $F(R)$ for a general λ $(0<\lambda<\infty)$ is shown in Fig. 2. It is positive for all $R<1$ and monotonically decreasing to zero at $R=1$. Thus Eq. (21) has a single fixed point at $R=1$ which implies that R will always converge to 1 regardless of the (fixed) value of λ. Near $R\sim 1$,

$$F(R\sim 1) \approx \frac{2}{3\pi}(1-R^2)^{3/2}, \tag{22}$$

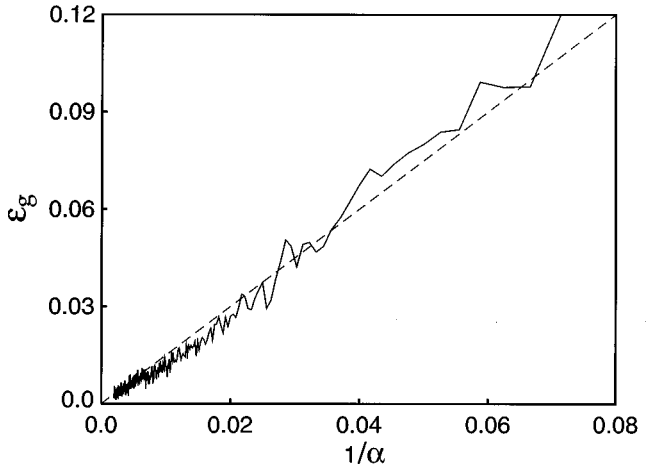

FIG. 3. Generalization error of a realizable perceptron vs the inverse of the number of examples per weight, α. The number of inputs is 50 and $\lambda=1$. The asymptotic convergence of the algorithm is compared with the theoretical prediction (dashed line) $\epsilon_g \approx 1.5/\alpha$.

independent of λ. Thus the asymptotic convergence of R to 1 is

$$\delta R = 1 - R \approx \left(\frac{3\pi}{2\sqrt{2}}\right)^2 \frac{1}{\alpha^2} \tag{23}$$

and ϵ_g vanishes as

$$\epsilon_g(\alpha) \approx \frac{3}{2\alpha}. \tag{24}$$

According to Eq. (23), the asymptotic behavior of ϵ_g is independent of λ, and the power law is in accordance with the general theory of OLGA.

The numerical simulation of this problem is shown in Fig. 3. Each component of the initial student weight vector is drawn randomly from a uniform distribution between -1.0 and 1.0 followed by a weight normalization, $\mathbf{w}\cdot\mathbf{w}=N$. As the figure shows, there is a very good agreement with the predictions of the theory.

C. Learning a perceptron rule with output noise

We now consider the case where the labels of the teacher perceptron are corrupted by a noise. The noise is generated uniformly with the probability p, $0<p<0.5$. The target rule is given as

$$\sigma_0(\mathbf{s}) = \begin{cases} +\mathrm{sgn}(\mathbf{w}_0\cdot\mathbf{s}) & \text{with probability } 1-p \\ -\mathrm{sgn}(\mathbf{w}_0\cdot\mathbf{s}) & \text{with probability } p. \end{cases}$$

Obviously, the optimal weight vector is still $\mathbf{w}=\mathbf{w}_0$ and $\epsilon_{\min}=p$. It has been shown previously that if the input distribution is isotropic the conventional on-line perceptron algorithm converges to the teacher weights even in the presence of output noise [15–17]. This is not the case for a general nonuniform input distribution. To demonstrate the advantage of the present algorithm we have considered the case of an input $\mathbf{s}$ which is randomly chosen from a Gaussian distribution with a nonzero mean,

$$\mathcal{P}(\mathbf{s}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}} e^{-(s_i-u_i)^2/2}, \tag{25}$$

with

$$Q_0 = \frac{1}{\sqrt{N}}\mathbf{u}\cdot\mathbf{w}_0, \quad 0<Q_0<1. \tag{26}$$

If the center vector $\mathbf{u}$ is neither parallel nor orthogonal to $\mathbf{w}_0$, the conventional on-line perceptron algorithm does not converge to $\mathbf{w}_0$ [15–17].

According to paper I, in the case of learning a realizable rule corrupted by output noise, OLGA converges to $\mathbf{w}_0$ in the limit of large λ. For large fixed λ ϵ_g deviates from its minimal value p by an amount of the order of $1/\sqrt{\lambda}$. Furthermore, when λ is made to increase with n, ϵ_g approaches p by a power law which in the optimal case ($\lambda \propto \lambda_0 n^2$) is inversely proportional with n. We first check these results using the large N analytic theory.

In the Appendix we show that $\epsilon_g(\mathbf{w})$ is given by

$$\epsilon_g(R,Q) = p + (1-2p)\left[\int_{-\infty}^{-Q} Dy H\left(\frac{-Q_0 - Ry}{\sqrt{1-R^2}}\right) + \int_{-Q}^{\infty} Dy H\left(\frac{Q_0 + Ry}{\sqrt{1-R^2}}\right)\right], \tag{27}$$

where Q is defined as

$$Q \equiv \frac{\mathbf{u}\cdot\mathbf{w}}{\sqrt{N}}. \tag{28}$$

As expected, $\epsilon_{\min}=p$ if $\mathbf{w}=\mathbf{w}_0$, i.e., $R=1$ and $Q=Q_0$.

1. Fixed large λ

In the Appendix we derive the mean-field equations for R and Q in the large N limit. It can be shown that for finite λ the fixed-point value of R is less than 1, but approaches 1 as $\lambda\to\infty$. In this limit we study the asymptotic behavior of R and Q in the vicinity of $R\sim 1$ and $Q\sim Q_0$. We define

$$\begin{aligned} r(\alpha) &= \lambda[1-R(\alpha)], \\ q(\alpha) &= \lambda[Q_0 - Q(\alpha)]. \end{aligned} \tag{29}$$

The equations for the scaled variables are of the form

$$\frac{dr}{d\alpha} = \lambda^{-1/2} f(r), \tag{30}$$

$$\frac{dq}{d\alpha} = \lambda^{-1/2} G(r,q), \tag{31}$$

where $f(r)$ and $G(r,q)$ are given in Eqs. (A11) and (A12). The shape of the function $f(r)$ is shown in Fig. 4. Because of the form of the above equations the equation that determines

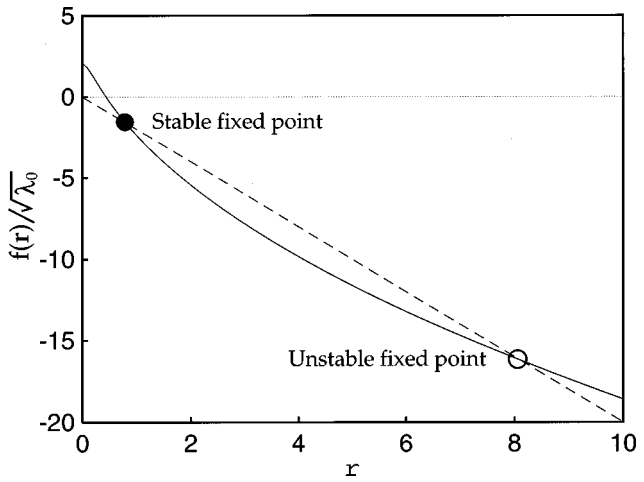

FIG. 4. The shape of $f(r)$ given by Eq. (A11) for $Q_0=0.5$, $p=0.2$, and $\lambda_0=0.001$. The solid line shows $f(r)/\sqrt{\lambda_0}$ and the dashed line is $-2r$. If $\lambda_0<\lambda_{\max}$, there are two fixed points, one stable (filled circle) and the other unstable (open circle), as shown where $f(r)/\sqrt{\lambda_0}=-2r$.

the convergence of the dynamics is Eq. (30). From the shape of $f(r)$, it is seen that the system converges to the root of f and G, given by r^* and q^*, where

$$f(r^*)=0, \tag{32}$$

$$G(r^*,q^*)=0. \tag{33}$$

This implies that at infinite time,

$$1-R=\frac{1}{\lambda}r^*, \quad \alpha\to\infty \tag{34}$$

$$Q_0-Q=\frac{1}{\lambda}q^*, \quad \alpha\to\infty. \tag{35}$$

Since from Eq. (27)

$$\epsilon_g-p\approx\frac{e^{-Q_0^2/2}}{\pi}(1-2p)\sqrt{2(1-R)}, \tag{36}$$

we obtain

$$\epsilon_g-p\approx\frac{e^{-Q_0^2/2}}{3\sqrt{\pi}}\frac{1}{\sqrt{\lambda}}, \quad \alpha\to\infty. \tag{37}$$

The convergence to this value is exponential in α.

2. Power-law schedule for λ

In order to achieve asymptotic convergence to $\epsilon_{\min}$, λ must be increased with time. We first consider a power-law schedule

$$\lambda(\alpha)=\lambda_0\alpha^{2\nu}. \tag{38}$$

Defining

$$r(\alpha)=\lambda(\alpha)[1-R(\alpha)] \tag{39}$$

and using Eq. (30) yields

$$\frac{dr}{d\alpha}=\frac{f(r)}{\sqrt{\lambda_0}\alpha^{\nu}}+\frac{2\nu r}{\alpha}. \tag{40}$$

It can be seen that convergence of r to a finite value requires (see paper I)

$$0\leqslant\nu\leqslant 1. \tag{41}$$

For $\nu<1$ the dominant term in the right hand side of Eq. (40) is the first term, yielding $r(\alpha)\to r^*$ as in the fixed λ case, Eq. (32). This implies that

$$\begin{aligned}\epsilon_g(\alpha)-p&\approx\frac{e^{-Q_0^2/2}}{3\sqrt{\pi}}\frac{1}{\sqrt{\lambda(\alpha)}}\\&\approx\frac{e^{-Q_0^2/2}}{3\sqrt{\pi\lambda_0}}\frac{1}{\alpha^{\nu}}.\end{aligned} \tag{42}$$

Note that the coefficient does not depend on the noise level p. For $\nu>1$ the dominant term in Eq. (40) is the second term, which results in the divergence of r with α. Optimal power of convergence is achieved for $\nu=1$ in which case the two terms in Eq. (40) contribute equally. In this case, if λ_0 is less than an upper bound $\lambda_{\max}$, $r(\alpha)$ converges to a value, a stable fixed point r^* as shown in Fig. 4, which depends on λ_0 (see the Appendix), and

$$\epsilon_g(\alpha)-p\propto(1-2p)\frac{e^{-Q_0^2/2}}{\pi}\sqrt{\frac{2r^*}{\lambda_0}}\frac{1}{\alpha}, \quad \nu=1, \quad \lambda_0<\lambda_{\max}. \tag{43}$$

If $\lambda_0>\lambda_{\max}$ and $\nu=1$, then the dominant term in Eq. (40) is the second term, which is

$$\frac{dr}{d\alpha}\approx\frac{2r}{\alpha}>0. \tag{44}$$

Therefore

$$r(\alpha)\propto\alpha^2 \tag{45}$$

and consequently

$$\delta R\propto\frac{1}{\lambda_0}, \quad \alpha\to\infty \tag{46}$$

and ϵ_g-p remains finite.

In Fig. 5 we show the simulation results for this problem with $N=50$, $p=0.2$, and $\lambda=10^{-5}t^2$. The results agree with the predicted inverse power-law learning curve. Inverse power-law convergence of ϵ_g, which is of course the optimal rate for this problem, is also obtained by other on-line algorithms for the perceptron [19,20]. As discussed in paper I, this rate of convergence is special for an unrealizability generated by uniform output noise. More generic unrealizable tasks are presented below.

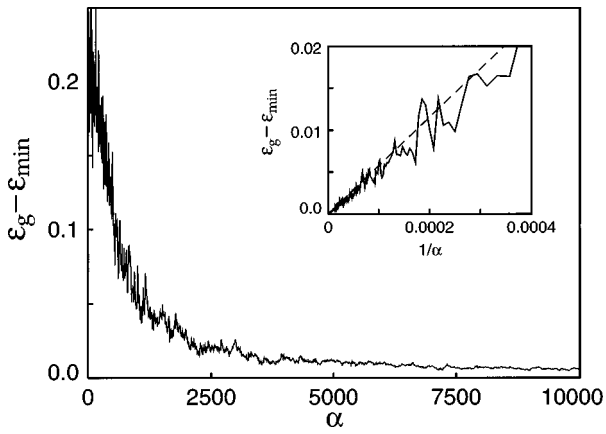

FIG. 5. Generalization error of a perceptron learning from examples generated by a perceptron with a uniform output noise, with a probability of mistake $p=0.2$. The number of inputs is 50. The input distribution is a Gaussian which is centered around a vector $\mathbf{u}$ with $\|\mathbf{u}\|=2$ and $\mathbf{u}\cdot\mathbf{w}_0=0.5\sqrt{N}$. The parameter λ is increased with time as $\lambda=\lambda_0\alpha^2$, with $\lambda_0=10^{-5}$. The inset compares the asymptotic $1/\alpha$ convergence of the algorithm with the theoretical prediction (dashed line).

D. Perceptron with Gaussian input noise

Another popular model of an unrealizable task is learning from examples in which the inputs $\mathbf{s}$ are corrupted by a noise η, i.e., the labels are given by

$$\sigma_0(\mathbf{s})=\mathrm{sgn}(\mathbf{w}_0\cdot\mathbf{s}+\eta) \tag{47}$$

and η is assumed to have a smooth distribution. This case falls under the category of a generic unrealizable rule. According to paper I OLGA should converge to the minimum of the generalization error in the limit of large λ. For an optimal schedule of λ ($\lambda=\lambda_0 n$) ϵ_g approaches $\epsilon_{\min}$ as the inverse of $\sqrt{n}$. We now study this model for a noise η that is generated randomly by a Gaussian distribution

$$D\eta=\frac{d\eta}{\sqrt{2\pi p^2}}\exp(-\eta^2/2p^2). \tag{48}$$

The distribution of each of the input components s_i is a Gaussian with zero mean and unit variance. We first discuss the analytic theory in the large N limit. The generalization error ϵ_g is given as

$$\epsilon_g(R)=2\int_{-\infty}^{+\infty}D\eta\int_0^{+\infty}DyH\left(\frac{Ry+p\eta}{\sqrt{1-R^2}}\right) \tag{49}$$

$$=\frac{1}{\pi}\arccos\left(\frac{R}{\sqrt{1+p^2}}\right). \tag{50}$$

From Eq. (50), we obtain for small $1-R$

$$\epsilon_g(R)\approx\epsilon_{\min}+\frac{1-R}{p\pi}, \tag{51}$$

with

$$\epsilon_{\min}=\epsilon_g(R=1)=\frac{1}{\pi}\arccos\left(\frac{1}{\sqrt{1+p^2}}\right). \tag{52}$$

In this case, the mean-field equation of R is

$$\frac{dR}{d\alpha}=F(R), \tag{53}$$

where

$$F(R)=-R\int_0^{\sqrt{2\lambda}}Dyy^2\int_{-\infty}^{+\infty}D\eta H\left(\frac{Ry+p\eta}{\sqrt{1-R^2}}\right)+\frac{(1-R^2)\sqrt{1-R^2+p^2}}{\pi(1+p^2)}\times\left[1-\exp\left(-\frac{(1+p^2)}{\lambda(1-R^2+p^2)}\right)\right]. \tag{54}$$

1. Fixed large λ

In contrast to the output noise case, the appropriate scaled variable in the limit of $\lambda\to\infty$ and $R\sim1$ is

$$r(\alpha)=\sqrt{\lambda}[1-R(\alpha)], \tag{55}$$

which obeys

$$\frac{dr}{d\alpha}\approx\frac{1}{\lambda}\left(\frac{1}{3\sqrt{\pi}}-\frac{2r}{p\pi}\right). \tag{56}$$

For fixed large λ, r converges exponentially to a fixed-point value $r^*=p\sqrt{\pi}/6$ yielding

$$\epsilon_g-\epsilon_{\min}\approx\frac{1}{6\sqrt{\pi}}\frac{1}{\sqrt{\lambda}}. \tag{57}$$

2. Power-law schedule for λ

In order to obtain that $\epsilon_g-\epsilon_{\min}$ vanishes in the limit of infinite time, we assume the power-law schedule

$$\lambda(\alpha)=\lambda_0\alpha^{\nu},\ \text{ where }\ 0<\nu\leqslant1 \tag{58}$$

and analyze the dynamics of $r(\alpha)=\sqrt{\lambda(\alpha)}[1-R(\alpha)]$. If $\nu<1$, the dominant terms in the differential equation for $r(\alpha)$ reduce to

$$\frac{dr}{d\alpha}\approx\frac{1}{\lambda_0\alpha^{\nu}}\left(\frac{1}{3\sqrt{\pi}}-\frac{2r}{\pi p}\right). \tag{59}$$

Thus δR vanishes as

$$\delta R(\alpha)\approx\frac{\sqrt{\pi}p}{6\sqrt{\lambda_0}}\alpha^{-\nu/2} \tag{60}$$

and ϵ_g converges to $\epsilon_{\min}$ as

$$\epsilon_g(\alpha)-\epsilon_{\min}\approx\frac{1}{6\sqrt{\pi\lambda_0}}\alpha^{-\nu/2}. \tag{61}$$

Note that the coefficient does not depend on a noise, p.

Optimal learning rate is obtained for $\nu=1$. In this case,

$$\frac{dr}{d\alpha}=\left(\frac{2}{\lambda_0\alpha}\right)\left[\frac{1}{6\sqrt{\pi}}-r\left(\frac{1}{\pi p}-\frac{\lambda_0}{4}\right)\right]. \tag{62}$$

Thus, in order to keep $r(\alpha)$ from growing, λ_0 has to satisfy

$$\lambda_0<\lambda_0^{\max}, \tag{63}$$

where $\lambda_0^{\max}=4/\pi p$. From Eq. (62), we obtain δR vanishing as $1/\sqrt{\alpha}$ which is

$$\delta R(\alpha)\approx\frac{2\sqrt{\pi}}{3(4-\pi p\lambda_0)}\frac{p}{\sqrt{\lambda_0\alpha}} \tag{64}$$

and ϵ_g converges as

$$\epsilon_g(\alpha)-\epsilon_{\min}\approx\frac{2}{3\sqrt{\pi\lambda_0}(4-\pi p\lambda_0)}\frac{1}{\sqrt{\alpha}}. \tag{65}$$

The optimal coefficient λ_0^* is

$$\lambda_0^*=\frac{4}{3\pi p}, \tag{66}$$

for which

$$\epsilon_g(\alpha)-\epsilon_{\min}\approx\frac{\sqrt{3p}}{8}\frac{1}{\sqrt{\alpha}}. \tag{67}$$

For $\nu=1$, if λ_0 does not satisfy Eq. (63), then the second term in Eq. (62) keeps growing while the first term remains finite. Thus Eq. (62) can be written approximately,

$$\frac{dr}{d\alpha}\approx\frac{r}{2\alpha}\left(1-\frac{\lambda_0^{\max}}{\lambda_0}\right). \tag{68}$$

Then δR vanishes as

$$\delta R(\alpha)\approx\frac{1}{\sqrt{\lambda_0}}\alpha^{-(\lambda_0^{\max}/2\lambda_0)} \tag{69}$$

and ϵ_g converges to $\epsilon_{\min}$ in suboptimal rate which is

$$\epsilon_g(\alpha)-\epsilon_{\min}\approx\frac{1}{\pi p\sqrt{\lambda_0}}\alpha^{-(\lambda_0^{\max}/2\lambda_0)}. \tag{70}$$

The results of the numerical simulations of the model and the theoretical asymptote are presented in Fig. 6.

E. Perceptron learning a committee-machine rule

Our final example of a perceptron learning an unrealizable rule is the case of a rule generated by a committee machine with three hidden units with weight vectors $\mathbf{w}_0^k$, $k=1,2,3$. Thus the output $\sigma_0(\mathbf{s})$ generated by the teacher is

$$\sigma_0(\mathbf{s})=\mathrm{sgn}\left(\sum_{k=1}^{3}\mathrm{sgn}(\mathbf{w}_0^k\cdot\mathbf{s})\right). \tag{71}$$

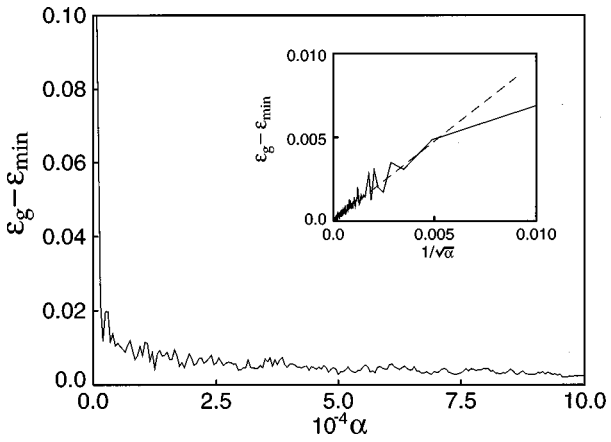

FIG. 6. Generalization error of a perceptron with input noise. The number of inputs is 50. The noise η is generated from a Gaussian distribution whose center is zero and standard deviation is 1.6. The parameter λ is increased with time as $\lambda=\lambda_0\alpha$, with $\lambda_0=0.01$. The inset compares the asymptotic convergence of the algorithm with theoretical prediction (dashed line).

We further assume that $\mathbf{w}_0^k\cdot\mathbf{w}_0^l=N\delta_{kl}$ where k, l denote the indices of hidden units and $\mathbf{w}_0^k\in R^N$.

From the analysis of paper I, we expect that the asymptotic behavior is similar to the case of a perceptron with input noise. In particular, for time-dependent λ the best convergence rate to $\epsilon_{\min}$ is achieved for $\lambda(\alpha)=\lambda_0\alpha$. In this case,

$$\epsilon_g(\alpha)-\epsilon_{\min}\propto\frac{1}{\sqrt{\alpha}},\alpha\to\infty \tag{72}$$

as in the case of input noise. This expectation is borne out by our numerical simulations, shown in Fig. 7. In the simulation the distribution of each of the input components s_i is a Gaussian with zero mean and unit variance. They demonstrate that with the above mentioned schedule for λ, the system converges to the minimum of ϵ_g with the rate of Eq.

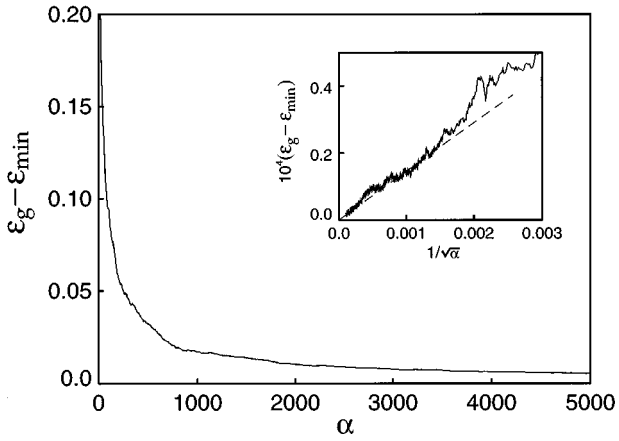

FIG. 7. Generalization error of a perceptron learning a rule generated by a three hidden unit committee machine with 50 inputs. The parameter λ is increased with time as $\lambda=\lambda_0\alpha$ with $\lambda_0=1.0$. The inset exhibits the asymptotic $1/\sqrt{\alpha}$ convergence of this algorithm. The dashed line is the best linear fit of the late part of the simulations.

(72). The analysis of these simulations depends on knowing with accuracy the value of the minimal generalization error. In the present case, ϵ_g is

$$\epsilon_g(\mathbf{w}) = 6\int_0^\infty Dx \int_0^\infty Dy H\left(\frac{R_1 x + R_2 y}{\sqrt{1-R_1^2-R_2^2}}\right) \tag{73}$$

$$-4\int_0^\infty Dx \int_0^\infty Dy \int_0^\infty Dz H\left(\frac{R_1 x + R_2 y + R_3 z}{\sqrt{1-R_1^2-R_2^2-R_3^2}}\right), \tag{74}$$

where R_k are the overlaps between $\mathbf{w}$ and $\mathbf{w}_0^k$ defined as

$$R_k = \frac{1}{N}\mathbf{w}\cdot\mathbf{w}_0^k. \tag{75}$$

The optimal value of ϵ_g is obtained when $\mathbf{w}$ is in the same distance from all three teacher weight vectors. Minimizing with respect to $R = R_k$ yields

$$R_k = \frac{1}{\sqrt{3}} \quad \text{for all} \quad k \tag{76}$$

and $\epsilon_{\min} = 0.162$.

IV. OLGA FOR A COMMITTEE MACHINE

A. Definition of the algorithm

For a learning system with a committee-machine architecture

$$\sigma(\{\mathbf{w}_l\},\mathbf{s}) = \mathrm{sgn}\left(\sum_{l=1}^{M} \mathrm{sgn}(\mathbf{w}_l\cdot\mathbf{s})\right), \tag{77}$$

the trained parameters are the M vectors $\mathbf{w}_l$, where M is an odd integer bigger than 1. We will assume that they are kept normalized so that $\mathbf{w}_l\cdot\mathbf{w}_l = N$ at all times. The on-line energy function E is

$$E(\{\mathbf{w}_l'\}|\{\mathbf{w}_l\}) = \epsilon(\{\mathbf{w}_l'\};\mathbf{s}) + \frac{\lambda}{2}\sum_{l=1}^{M} \|\mathbf{w}_l' - \mathbf{w}_l\|^2, \tag{78}$$

where $\{\mathbf{w}_l'\} = \{\mathbf{w}_l^n\}$ and $\{\mathbf{w}_l\} = \{\mathbf{w}_l^{n-1}\}$ are the new and current weight vectors of the student, respectively. After each presentation of a new labeled example, based on the general steps outlined in Sec. II, the learning algorithm is as follows.

(1) If $\sigma = \sigma_0$, $\mathbf{w}' = \mathbf{w}$.

(2) If $\sigma \neq \sigma_0$, then measure $\{h_l\}$ where

$$h_l \equiv \mathbf{w}_l^{n-1}\cdot\hat{\mathbf{s}}^n. \tag{79}$$

Note that since $\sigma \neq \sigma_0$ the number M' of local fields which has ''wrong outputs,'' i.e., $\sigma_0^n h_l < 0$ obeys $(M+1)/2 < M' \leq M$.

(3) Order the hidden units with $\sigma_0^n h_l < 0$ according to the magnitude of $|h_l|$ so that $|h_l| < |h_{l+1}|$. The candidates for updating are the units $l = 1, \ldots, M_{\text{update}}$ where

$$M_{\text{update}} = M' - (M-1)/2, \tag{80}$$

which is the minimal number of hidden units that need to be corrected in order to change the sign of σ.

(4) To minimize the change which will correct the error, each of the corresponding weight vectors $\mathbf{w}_l, l = 1, \ldots, M_{\text{update}}$ is updated according to the perceptron updating rule,

$$\mathbf{w}_l^n = A_l(\mathbf{w}_l^{n-1} - h_l\hat{\mathbf{s}}^n), \tag{81}$$

where $A_l = (1 - N^{-1}h_l^2)^{-1/2}$.

(5) The above rule is implemented only if the local fields of the candidates satisfy the bounded-change condition, which in the present case reduces to

$$\sum_{l=1}^{M_{\text{update}}} [1 - \sqrt{1-(h_l^2/N)}] < 1/\lambda N. \tag{82}$$

If the bounded-change condition is not satisfied, $\{\mathbf{w}_l'\} = \{\mathbf{w}_l\}$. For large N, the bounded-change condition can be simplified to

$$\sum_{l=1}^{M_{\text{update}}} h_l^2 < 2/\lambda, \tag{83}$$

see Eq. (12).

In the following we present our main numerical results for the convergence of this algorithm. All the simulations are performed with inputs $\mathbf{s}$ that are drawn from Gaussian distribution in which each component has a unit variance and zero mean. The weight vectors of the teacher are chosen orthogonal to each other, i.e., $\mathbf{w}_{0,k}\cdot\mathbf{w}_{0,l} = N\delta_{kl}$. The initial student vectors are generated randomly from a uniform distribution for each component of $\mathbf{w}_l$ between -1 and 1, followed by a normalization $\mathbf{w}_l\cdot\mathbf{w}_l = N$.

B. Realizable committee machine

Our first example is the case where the teacher has the same committee-machine architecture as the student, with M orthogonal weight vectors $\mathbf{w}_l^0$. The optimal value of ϵ_g is zero when $\{\mathbf{w}_l\} = \{\mathbf{w}_l^0\}$. Our numerical simulation results for $M = 7$ and $M = 19$ are shown in Figs. 8 and 9, respectively. The main result is that in all cases studied (different values of M, ranging between 3 and 19) the system converges to the global minimum of ϵ_g and the asymptotic convergence rate to zero ϵ_g is

$$\epsilon_g \approx \frac{1}{\alpha}, \tag{84}$$

where α is defined as the number of examples per weight. This convergence is identical with the result for the realizable perceptron case. As in the perceptron case, the above convergence is achieved even when λ is fixed in time. However, unlike the perceptron case, we find here that only when the initial student weight vectors are in the neighborhood of the teacher weight vectors ϵ_g converges to zero for all values of λ. To guarantee convergence from most (randomly sampled) initial conditions the constant λ may need to be larger than some lower bound (which may depend on M). For instance, for $M = 3$, we find that ϵ_g converges from most

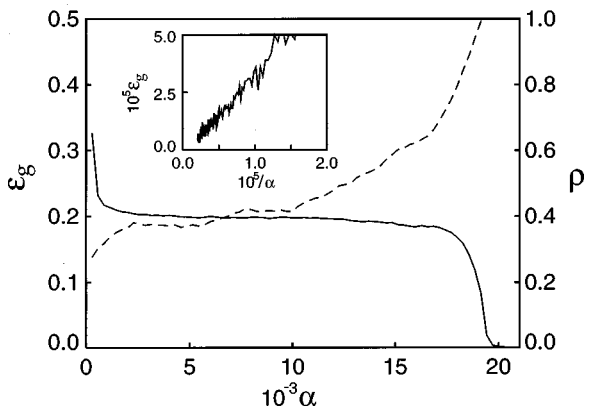

FIG. 8. Generalization error of a committee machine with $M=7$ vs the number of examples per weight, α. The number of inputs is 150 and $\lambda=200$. The dashed line indicates the corresponding order parameter ρ, Eq. (85). Inset exhibits the asymptotic convergence.

initial conditions to zero apparently for all values of λ. However, for $M \geqslant 5$, we find that λ should be at least of $O(100)$ to converge to $\epsilon_g = 0$. Otherwise depending on the initial conditions the system may get stuck in a suboptimal region of the weight space. This difference is indicative of the more complex structure of the learning dynamics in the committee-machine case.

Another indication of this complexity is the existence of plateaus in the curve of ϵ_g as shown in Figs. 8 and 9 for $M=7$ and $M=9$, respectively. An interpretation of this plateau in terms of spurious fixed points of the mean-field deterministic dynamics is not clear, since as far as we can tell the macroscopic order parameters keep changing with time although very little change happens in the value of ϵ_g. To illustrate this important phenomenon we define an effective order parameter ρ as

$$\rho = 1/M \sum_l R_{k_0,l}, \tag{85}$$

where $R_{k,l}$ is the overlap between the teacher's kth weight vector and the student's lth vector, $R_{k,l} = \mathbf{w}_{0,k} \cdot \mathbf{w}_k / N$, and $R_{k_0,l}$ denotes $\mathrm{Max}\{R_{1,l}, R_{2,l} \ldots, R_{M,l}\}$. The temporal evolution of ρ alongside that of ϵ_g is shown in Figs. 8 and 9. These

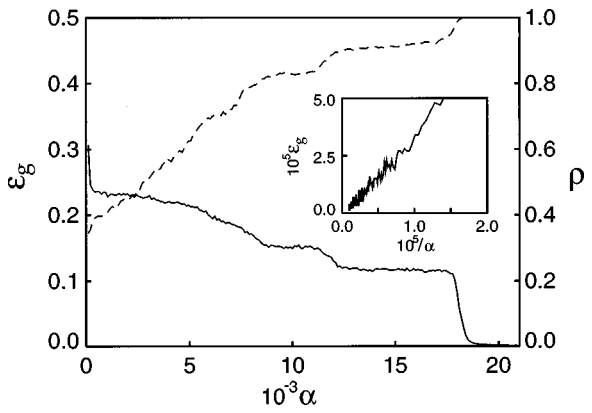

FIG. 9. The same as in Fig. 8 but with $M=19$, the number of inputs is 50 and $\lambda=100$.

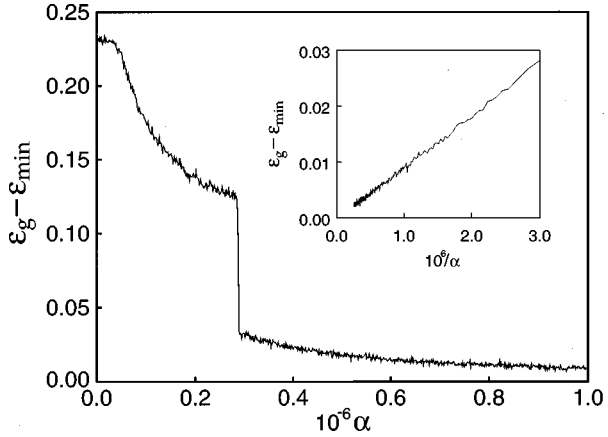

FIG. 10. Generalization error of a committee machine with $M=3$ learning from examples generated by a teacher with the same architecture but corrupted by uniform output noise, with a probability of mistake $p=0.2$. The number of inputs is 50. The parameter λ is increased with time as $\lambda = \lambda_0 \alpha^2$, with $\lambda_0 = 9.0 \times 10^{-10}$. The inset exhibits the asymptotic $1/\alpha$ convergence.

results show that ρ changes substantially even in the plateaus of ϵ_g. Thus these plateaus do not correspond to genuine fixed points (even in the large N limit). Similar phenomena have been observed and discussed in other on-line learning algorithms for ''soft'' committee machines [9,12].

C. Committee machine with output noise

As an example for learning unrealizable tasks, we tried output noise for a committee machine. The teacher's output is corrupted by a noise generated uniformly with probability p. The output of the teacher is given as

$$\sigma_0 = \begin{cases} +\mathrm{sgn}\left(\sum_{l=1}^{M_0} \mathrm{sgn}(\mathbf{w}_{0,l} \cdot \mathbf{s}) \right) & \text{with probability } 1-p \\ -\mathrm{sgn}\left(\sum_{l=1}^{M_0} \mathrm{sgn}(\mathbf{w}_{0,l} \cdot \mathbf{s}) \right) & \text{with probability } p. \end{cases}$$

In this problem, $\epsilon_{\min} = p$, and the optimal weight vectors are $\{\mathbf{w}_l\} = \{\mathbf{w}_{0,l}\}$.

We performed numerical simulations of this problem with $M=3$ and $N=50$, and various values of the (fixed) parameter λ. We find (results not shown) a residual ϵ_g which for large λ behaves as

$$\epsilon_g - p \approx 1/\sqrt{\lambda}. \tag{86}$$

With a power law λ we obtained our best results by choosing $\lambda(\alpha)$ as

$$\lambda = \lambda_0 \alpha^2, \tag{87}$$

which yielded

$$\epsilon_g - p \approx 1/\alpha \tag{88}$$

as shown in Fig. 10.

The results exhibit a remarkably sharp drop in ϵ_g at $\alpha \approx 2.9 \times 10^5$. Such a sharp change in ϵ_g was not observed in any of the many systems and target rules that we have simu-

lated. Finally, our simulations showed that in order to guarantee convergence of ϵ_g to $\epsilon_{\min}$ from most of the initial conditions λ_0 must not be greater than $\approx 10^{-10}$. For larger values of λ_0 the system may converge to a spurious fixed point depending on the initial conditions. If $\lambda(\alpha)$ is chosen to increase less rapidly than α^2 convergence to $\epsilon_{\min}$ is insensitive to the prefactor. This behavior, which is similar to that shown above in the analytical solution of a single-layer perceptron with output noise [Eqs. (42)–(46)], is in accordance with the general theory of OLGA in the case of output noise, and will be further discussed in the last section.

V. OLGA FOR A WINNER-TAKES-ALL CLASSIFIER

A. Definition of algorithm

In this section we consider a winner-takes-all classifier which can be modeled by a two-layer neural network with a layer of M linear hidden units. The weights $\{\mathbf{w}_l\}$ from the input to hidden units are learned. Each hidden unit is connected to its output unit by a weight fixed as one. The output of the system is the index of hidden unit, l with maximum local field h_l, which is given as

$$\sigma(\mathbf{s}) = \mathrm{argmax}_l\{\mathbf{w}_l \cdot \mathbf{s}\}. \tag{89}$$

The number of units in the hidden layer is arbitrary. We will adopt an overall weight normalization

$$\sum_{l=1}^{M} \|\mathbf{w}_l\|^2 = MN. \tag{90}$$

Finally we assume that the labels on the examples are themselves generated by a classifier so that $\sigma_0 = 1, \ldots, M$.

For the sake of simplicity we first define OLGA for a WTA with $M=3$. Let us assume that a new example $\mathbf{s}^n$ generates the following local fields:

$$h_l \equiv \mathbf{w}_l^{n-1} \cdot \hat{\mathbf{s}}^n, \tag{91}$$

where as before $\hat{\mathbf{s}}$ is unit vector in the direction of $\mathbf{s}$. Suppose that the winner among these units is k, i.e., $\sigma(\mathbf{s}^n) = k$ and the label on this example is $\sigma_0(\mathbf{s}^n) = k_0$. We need to consider only the case $k \neq k_0$, otherwise we keep the current weight vectors. Let us assume that, say, $k=1$ and $k_0 = 3$. This means that $h_1 > h_2, h_3$ and we want to make a minimal change in the weights so that the new local field h_3' will be the largest field. We have to consider separately the following two cases.

(1) $h_2 < (h_1 + h_3)/2$. In this case we have to increase h_3 and to decrease h_1 by the same amount so that

$$h_1' = h_3' = \frac{h_1 + h_3}{2} \equiv h'. \tag{92}$$

The local field h_2 need not be changed, since the above change will leave h_2 smaller than h' as desired. The updating rule which satisfies Eq. (92) is

$$\mathbf{w}_l^n = A[\mathbf{w}_l^{n-1} + (h' - h_l)\hat{\mathbf{s}}^n] \tag{93}$$

for $l = 1$ and 3, whereas

$$\mathbf{w}_2^n = A\mathbf{w}_2^{n-1}. \tag{94}$$

The coefficient A is determined by the normalization condition Eq. (90). The bounded change condition is given as

$$\sum_{l=1}^{3} \|\mathbf{w}_l^n - \mathbf{w}_l^{n-1}\|^2 < 2/\lambda. \tag{95}$$

If this condition holds, we accept the new weight vectors. Otherwise, we keep the current weight vectors $\{\mathbf{w}_l^{n-1}\}$.

(2) $h_2 > (h_1 + h_3)/2$. This can occur only if $h_3 < h_2 < h_1$. In order to correct the instantaneous error it is necessary to update all three hidden units so that the new local fields $\{h_l'\}$ should satisfy the condition

$$h_1' = h_2' = h_3' = \frac{h_1 + h_2 + h_3}{3} \equiv h'. \tag{96}$$

The updating rule for each weight vector which leads to Eq. (96) is given by Eq. (93) for all three vectors. This update is performed provided the bounded change condition Eq. (95) is satisfied.

Extending this algorithm for an arbitrary number of hidden units is straightforward. We again denote the label of the new input as k_0 and assume that it does not agree with the winner of the current weights. The hidden units that are candidates for weight change are those with local fields that are greater than h_{k_0}. We order these units so that $h_l < h_{l-1}$ for $l = 1,2, \ldots, k_0$. Using this rearrangement of indices, $\sigma(\mathbf{s}^n) = 1$ whereas $\sigma_0(\mathbf{s}^n) = k_0$. In order to determine how many of these k_0 units should be actually updated we proceed as follows. We first consider the possibility that only h_1 and h_{k_0} need to be changed, in which case the change will be given by Eq. (93) with $h' = (h_1 + h_{k_0})/2$. We thus compare h_2 to this value of h'. If $h_2 < h'$, we accept the new weight vectors $\mathbf{w}_1^n$ and $\mathbf{w}_{k_0}^n$ and for the rest, we multiply the current weight vectors by the normalization factor A, i.e., $A\mathbf{w}_2^{n-1}, A\mathbf{w}_3^{n-1}, \ldots, A\mathbf{w}_{k_0-1}^{n-1}$. If $h_2 > h'$, then the next candidate vectors are $\mathbf{w}_1^n$, $\mathbf{w}_2^n$, and $\mathbf{w}_{k_0}^n$. The corresponding new local fields are $h' = (h_1 + h_2 + h_{k_0})/3$. Now we compare h_3 with this value of h'. If $h_3 < h'$, we stop searching for a candidate for updating and the new vectors are $\mathbf{w}_1^n$, $\mathbf{w}_2^n$, and $\mathbf{w}_{k_0}^n$ given by Eq. (93) and the rest of weight vectors are multiplied by the normalization factor A. Otherwise, we continue this procedure as described above until we find a unit with an index L which satisfies $h_{L+1} \leqslant h'$, where

$$h' = \frac{\sum_{l=1}^{L} h_l + h_{k_0}}{L+1}. \tag{97}$$

The associated weight vector updates of all the units $h_1, \ldots, h_L, h_{k_0}$ are as in Eq. (93) and the rest are obtained by $\mathbf{w}_l^n = A\mathbf{w}_l^{n-1}$. Finally, we check the bounded-change condition,

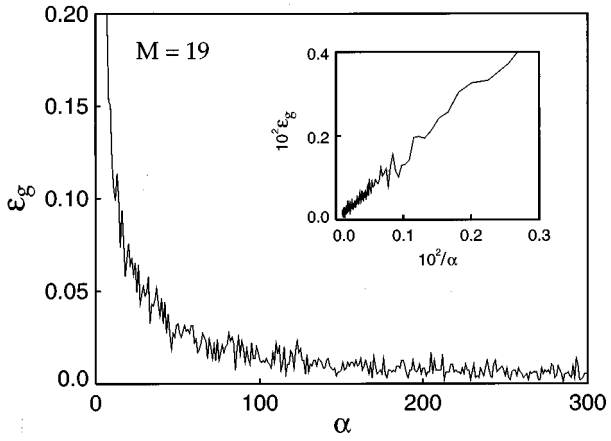

FIG. 11. Generalization error of a WTA with $M=19$ learning a rule generated by the same architecture. The number of inputs is 150 and $\lambda=1$. The inset exhibits the asymptotic $1/\alpha$ convergence of this algorithm.

$$\sum_{l=1}^{M} \|\mathbf{w}_l^n - \mathbf{w}_l^{n-1}\|^2 < 2/\lambda. \tag{98}$$

If this condition does not hold for the above update we keep the current weight vectors $\{\mathbf{w}_l^{n-1}\}$.

B. Realizable WTA

We have applied the above algorithm to simulate the case of a WTA learning examples generated by a teacher WTA, both having the same number of hidden units M. The weight vectors of the teacher are chosen orthogonal to each other, i.e., $\mathbf{w}_{0,k}\cdot\mathbf{w}_{0,l}=N\delta_{k,l}$. The initial value of each student weight is drawn from a uniform distribution of between -1 and $+1$. The weight vectors are then normalized so that $\Sigma_{l=1}^{M}\|\mathbf{w}_l\|^2=MN$. Each component of the input $\mathbf{s}$ was drawn from a Gaussian distribution with zero mean and unit variance. The numerical simulations used a fixed value of λ and M ranged between 3 and 19. An example of the simulations is shown in Fig. 11. The main result is that in all cases studied, the system converges to $\epsilon_g=0$ and the asymptotic convergence rate to zero ϵ_g is

$$\epsilon_g \approx \frac{1}{\alpha}, \tag{99}$$

where α is the number of examples per weight. The behavior of this system is similar to that of the realizable perceptron shown in Sec. III B. Not only is the asymptotic convergence the same, but also the overall behavior of ϵ_g. Thus in both cases ϵ_g decreases smoothly and rapidly from most initial conditions and this behavior does not seem to require particularly large λ. This should be contrasted with the case of the realizable committee machine discussed in Sec. IV B above.

VI. SUMMARY AND DISCUSSION

In this paper we derived explicit update rules that follow from OLGA for a single-layer perceptron, a two-layer committee machine, and a WTA classifier. We have studied the convergence properties of the update rules for a variety of target rules, both realizable and unrealizable. The analytical results (for the single-layer perceptron) as well as the extensive numerical simulations confirm the main general predictions of the asymptotic convergence of the deterministic limit of OLGA that were derived in paper I.

The OLGA update rule for a single-layer perceptron is related to other learning algorithms which were proposed for the perceptron. In particular, if we take the zero λ limit of the OLGA update rule, Eq. (9), and drop the normalization condition, we obtain the simple update rule

$$\mathbf{w}^n = \mathbf{w}^{n-1} - h_{n-1}\hat{\mathbf{s}}^n, \quad \sigma_0 h_{n-1} < 0, \tag{100}$$

and $\mathbf{w}^n=\mathbf{w}^{n-1}$ if $\sigma_0 h_{n-1}>0$. This update rule is one variant of the ''orthogonal projection'' algorithm proposed as early as 1954 [18,21,22] for learning a set of linear inequalities. More recently it was studied under the name ''AdaTron algorithm'' [23]. The original analysis of the algorithm was performed in the context of batch learning. More recently, the performance of the AdaTron as an on-line algorithm for a perceptron has been studied [24] analytically for a perceptron learning a realizable rule with Gaussian distributed inputs, using mean-field theory, similar to our treatment of Sec. III B. Our prediction of the asymptotic convergence of this case, Eq. (24), agrees with the previous analysis [24] (see also Ref. [13]).

It should be emphasized that the AdaTron on-line algorithm is adequate only if the data are generated by a perceptron. Our work focuses mostly on the problem of on-line learning of unrealizable rules. In such cases, the AdaTron algorithm does not converge in general to the minimum of ϵ_g even in the local sense. To ensure local convergence to the minimum of ϵ_g one must use OLGA with the constraint imposed by λ on the above update with λ that grows with time, as was shown in this paper.

Considering the update rule for the committee machine, it should be noted that unlike the backpropagation algorithm, OLGA's update rule is nonlocal. Although each updating hidden unit follows a perceptron update rule for its weight vector, only part of the units update their weights at each time step. The decision whether to update at all (i.e., the bounded change criterion) as well as the choice of the updating units depend on the local fields of all the units, as follows from Sec. IV A. A similar nonlocality occurs for the WTA network, Sec. V A. Deriving efficient OLGA update rules for networks with more hidden layers is an interesting issue beyond the scope of the present study [25].

The present paper and paper I show that the convergence properties of OLGA in discrete output systems are similar to those of the stochastic gradient descent algorithm in smooth systems. Both algorithms contain a learning rate parameter which controls the size of the changes made at each step. When the learning rates are small the two algorithms spend most of the time in the vicinity of the local minima of the generalization error. Furthermore, by an appropriate power-law decrease of the learning rate these local minima become the true fixed points of the dynamics.

Both the on-line gradient descent algorithm and OLGA suffer from the same fundamental weakness. They do not guarantee convergence to the global minimum of ϵ_g. Guar-

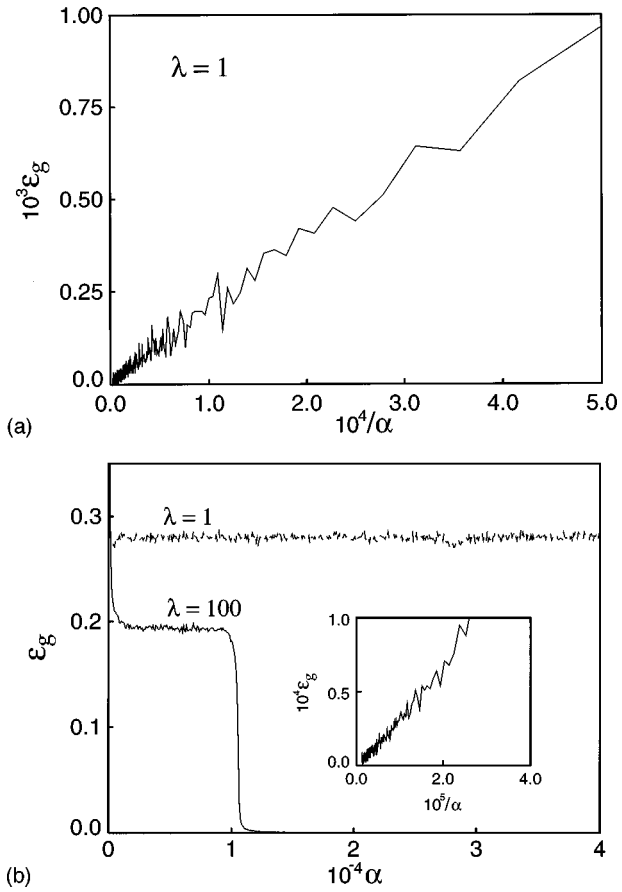

FIG. 12. Demonstration of the difference between the local and global convergence of OLGA in the case of a committee machine with five hidden units learning a rule generated by the same architecture. The number of inputs is 150. (a) The asymptotic $1/\alpha$ convergence of generalization error with $\lambda=1$ when the initial weights of the student network are close to the teacher's weights (the initial overlaps of the corresponding student and teacher weight vectors were chosen to be 0.8). (b) Generalization error when the initial weights are far away from the teacher's weights. With $\lambda=1$, the algorithm does not converge to the global minimum. With $\lambda=100$, the algorithm converges to the global minimum. The inset exhibits the asymptotic $1/\alpha$ convergence with $\lambda=100$.

anteeing global convergence requires in both cases the addition of stochastic noise in the learning dynamics. In the case of the stochastic gradient descent algorithm this amounts to using a (discrete time) Langevin dynamics, as studied in detail by Kushner [26]. In the case of OLGA, one needs to use OLGA at finite temperature, as discussed in detail in paper I. The difference between local and global convergence of OLGA has been demonstrated here in the case of a committee machine with five hidden units learning a rule generated by the same architecture. If the initial weights of the student network are close to the teacher's weights, the system converges to the teacher's weights even when λ is small [see Fig. 12(a)] as predicted for a general realizable rule in paper I. However, if the initial weights are far away from the global minimum, we find numerically that only when $\lambda>100$ that system converges to the global minimum [see Fig. 12(b)].

An important condition that a viable learning algorithm must satisfy is that its applicability should be largely independent of the nature of the target rule or the distribution of inputs. The results of the two papers demonstrate that this is indeed the case with OLGA. We have shown here that the update rules derived from OLGA vary according to the architecture of the learning system. However, they are essentially independent of the target rules or the specific form of the input distribution. This is similar to the stochastic gradient descent algorithm. Evaluating the local gradient depends on the system's architecture but not on the target rule. Furthermore, no parameter has to be especially tuned in order to guarantee convergence to a (local) minimum of ϵ_g. In particular, for a generic unrealizable rule if one chooses a power-law increase of λ with a suboptimal power, i.e., $\lambda=\lambda_0 n^\nu$ with $0<\nu<1$ local convergence to a minimum of ϵ_g is guaranteed for all values of the coefficient λ_0. To emphasize the important universality of OLGA we present in Fig. 13 the simulation results of OLGA for a single-layer perceptron learning a variety of unrealizable rules. In all cases we used the same schedule for λ, namely, $\lambda=0.01\lambda^{3/4}$. As the figure shows, not only does the system converge to $\epsilon_{\min}$ in all the cases but all the learning curves exhibit the same power-law convergence, namely, $\epsilon_g-\epsilon_{\min}\propto 1/\sqrt{\lambda}$.

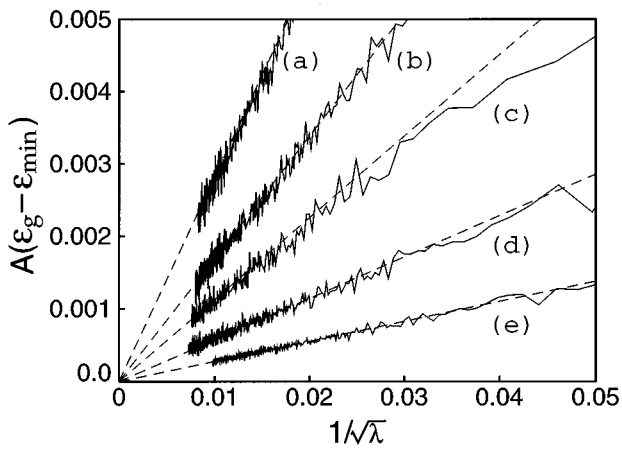

FIG. 13. ϵ_g of OLGA for a perceptron learning: a perceptron rule with output noise with noise levels p, $p=0.3$ (a), 0.4 (b), and 0.48 (c); a perceptron rule with a Gaussian input noise with standard deviation 1.6 (d); a committee-machine rule with three hidden units (e). The simulations (solid lines) are with 50 inputs, each has zero mean and a unit variance Gaussian distribution. The schedule of λ is $\lambda=0.01\,\alpha^{3/4}$. The dashed lines are theoretical asymptotes except for (e), where it is the best linear fit. The theoretically evaluated $\epsilon_{\min}$ is p (a)–(c); 0.3222 (d); 0.1623 (e). The factor $1/A$ in the vertical axis is 1.5, 0.9, 0.6, 0.6, and 0.3, for (a)–(e), respectively.

If optimal power-law convergence is sought then in the generic unrealizable case, one needs to choose $\lambda=\lambda_0 n$ with λ_0 smaller than an upper cutoff $\lambda_0^{\max}$. In general this cutoff is unknown, since it depends on the target rule and input distribution. Furthermore, even in this case if $\lambda_0>\lambda_0^{\max}$, the system still converges to the local minimum of ϵ_g though with a slower rate, as shown in the case of perceptron learning data corrupted by input noise [see Eqs. (68)–(70)]. These properties are completely analogous to the behavior of the on-line gradient descent algorithm with a power-law decrease of the learning rate. An exception is the case of output

noise. Here one can achieve a considerably faster rate of convergence since the value of ν can be as large as 2. In this case, if one chooses $\lambda=\lambda_0 n^2$ with $\lambda_0>\lambda_0^{\max}$ where $\lambda_0^{\max}$ is system dependent then the system will not converge to the local minimum of ϵ_g as we show analytically in the case of a single-layer perceptron, Eq. (46) above.

The above property of OLGA should be contrasted with the recent introduction of the so-called ''optimal on-line learning'' [27–30]. In this approach, the generalization error as a function of the student's weights is calculated using mean-field theory. Minimizing this function yields update rules that are optimal in the sense that they generate the fastest rate of reduction in ϵ_g. These calculations are interesting in that they provide a lower bound on the performance of on-line learning algorithms. However, they cannot be considered as interesting models of on-line learning. This is because of the obvious reason that the calculation of ϵ_g and its derivatives is possible only in those cases where the nature of the target rule is known as well as that of the input distribution. Even under these conditions, the mean-field calculations can be performed only in a restricted class of problems. It is therefore unfortunate that optimal on-line learning calculations are portrayed as on-line learning *algorithms* (see Refs. [31] and [32]). Regrettably, misconceptions about the role of analytical solutions of synthetic models are not uncommon in the field of neural networks. The need to gain theoretical insight into the performance of a model may require investigating its behavior in artificially simplified conditions. However, in order for the model to be interesting at all, it should be well defined and applicable under realistic conditions.

The advantages of using OLGA in real world problems have to be judged relative to other commonly used supervised on-line learning algorithms with similar network architectures, notably the backpropagation algorithm. The main advantage of OLGA over the back-propagation algorithm is in problems where the desired output is discrete, such as in learning Boolean functions, or classification tasks. Backpropagation, being a gradient-based method, requires that all the neurons (except possibly the input ones) have smooth outputs. Applying the backpropagation algorithm to these problems requires adding *ad hoc* parameters such as the form of the sigmoidal input-output functions of the neurons and their gains. In addition, the smooth error function used (implicitly) in the training is different from the discrete measure of error used in gauging the performance of the network after training. In contrast, algorithms such as OLGA allow us to keep the discrete nature of the outputs (and possibly also hidden units) and are thus simpler to interpret in terms of the underlying tasks. Whether this formal advantage can be translated to practical advantages in learning classification tasks remains to be studied.

In conclusion, we note that several important questions regarding on-line learning are still open. Among them is the entire issue of the global convergence properties of on-line learning discussed above. Another issue is the on-line learning in systems with discrete valued parameters, such as networks with binary weights. Most on-line algorithms including OLGA rely on the possibility of making small changes in a single update hence they are inadequate for discrete valued weights.

ACKNOWLEDGMENTS

We thank Y. Freund and R. Schapire for illuminating discussions about on-line learning, which led us to the construction of OLGA. We also thank N. Barkai, M. Kearns, D. Lee, S. Seung, and C. A. van Vreeswijk for very helpful discussions about OLGA and other on-line learning algorithms. We are grateful to Hyoungsoo Yoon for helpful discussions on perceptron learning algorithms. This research is partially supported by a grant from the United States–Israel Binational Science Foundation (BSF), Jerusalem, Israel.

APPENDIX: MEAN-FIELD THEORY OF PERCEPTRON WITH OUTPUT NOISE

The definition of the generalization is

$$\epsilon_g(\mathbf{w})=\langle\!\langle\langle\Theta(-\sigma_0(\mathbf{s})\sigma(\mathbf{w}\cdot\mathbf{s}))\rangle\!\rangle\!\rangle, \tag{A1}$$

where $\langle\!\langle\ \rangle\!\rangle$ denotes average over the input distribution $\mathcal{P}(\mathbf{s})$ given by Eq. (25) and $\langle\ \rangle$ denotes here average over the output noise. Performing the average over the output noise we obtain

$$\epsilon_g(\mathbf{w})=(1-2p)\langle\!\langle\Theta(-\sigma_0(\mathbf{s})\mathrm{sgn}(\mathbf{w}\cdot\mathbf{s}))\rangle\!\rangle+p. \tag{A2}$$

Performing the average over the input distribution yields

$$\epsilon_g(R,Q)=p+(1-2p)\left[\int_{-\infty}^{-Q}DyH\left(\frac{-Q_0-Ry}{\sqrt{1-R^2}}\right)+\int_{-Q}^{\infty}DyH\left(\frac{Q_0+Ry}{\sqrt{1-R^2}}\right)\right], \tag{A3}$$

where Q and Q_0 are defined in Eqs. (28) and (26).

In the large N limit R and Q obey the differential equations

$$\frac{dR}{d\alpha}=\frac{R}{2}\langle h^2\rangle_D-\langle h^0 h\rangle_D,$$
$$\frac{dQ}{d\alpha}=\frac{Q}{2}\langle h^2\rangle_D-\langle(\mathbf{u}\cdot\mathbf{s})h\rangle_D, \tag{A4}$$

where the subscript D means averaging over the volume of $\mathbf{s}$ which satisfies the bounded-change condition, Eq. (12), and $\alpha=n/N$ is the scaled time variable. Performing the average over the inputs, we obtain

$$\frac{dR}{d\alpha}=-(1-2p)[f_1(R,Q_0,Q)+f_1(R,-Q_0,-Q)]-pf_2(R,Q_0,Q), \tag{A5}$$

where

$$f_1(R,Q_0,Q)=\frac{1}{\sqrt{2\pi}}\int_{-\sqrt{2/\lambda}}^{0}dy e^{-(y-Q)^2/2}\times\left[\left(\frac{R}{2}y^2+Ay\right)H\left(\frac{-Ry-A}{\sqrt{1-R^2}}\right)+y\sqrt{\frac{1-R^2}{2\pi}}\exp\left(-\frac{(Ry+A)^2}{2(1-R^2)}\right)\right] \tag{A6}$$

and

$$f_2(R,Q_0,Q)=\frac{1}{\sqrt{2\pi}}\int_{-\sqrt{2/\lambda}}^{+\sqrt{2/\lambda}}dy e^{-(y-Q)^2/2}\left(\frac{R}{2}y^2+Ay\right), \tag{A7}$$

$$\frac{dQ}{d\alpha}=-(1-2p)[f_3(R,Q_0,Q)-f_3(R,-Q_0,-Q)]-pf_4(R,Q_0,Q), \tag{A8}$$

where

$$f_3(R,Q_0,Q)=\frac{1}{\sqrt{2\pi}}\int_{-\sqrt{2/\lambda}}^{0}dy e^{-(y-Q)^2/2}\times\left[\left(\frac{Q}{2}y^2+(\mathbf{u}^2-Q^2)y\right)H\left(\frac{-Ry-A}{\sqrt{1-R^2}}\right)+\frac{Ay}{\sqrt{2\pi(1-R^2)}}\exp\left(-\frac{(Ry+A)^2}{2(1-R^2)}\right)\right] \tag{A9}$$

and

$$f_4(R,Q_0,Q)=\frac{1}{\sqrt{2\pi}}\int_{-\sqrt{2/\lambda}}^{+\sqrt{2/\lambda}}dy e^{-(y-Q)^2/2}\times\left(\frac{Q}{2}y^2+(\mathbf{u}^2-Q^2)y\right). \tag{A10}$$

Here $y=h$, $A\equiv Q_0-QR$, and $\mathbf{u}^2=\mathbf{u}\cdot\mathbf{u}$. The above equations hold for arbitrary value of λ. To derive the large λ limit, we scale the variables R and Q as in Eqs. (29) and expand Eqs. (A5) and (A8) up to $O(1/\lambda^{3/2})$. This results in Eqs. (30) and (31), for $r(\alpha)$ and $q(\alpha)$, respectively, with

$$f(r)=\frac{2}{\sqrt{\pi}}e^{-Q_0^2/2}\left[\frac{p}{3}+(1-2p)\left\{\int_0^1 dy y^2 H\left(\frac{y}{\sqrt{r}}\right)-\sqrt{\frac{2}{\pi}}r^{3/2}(1-e^{-1/2r})\right\}\right], \tag{A11}$$

$$G(r,q)=\frac{2}{\sqrt{\pi}}e^{-Q_0^2/2}\left\{Q_0(1+2\mathbf{u}^2-2Q_0^2)\times\left[\frac{p}{3}+(1-2p)\int_0^1 dy y^2 H\left(\frac{y}{\sqrt{r}}\right)\right]-\frac{(1-2p)}{\sqrt{2\pi}}(1+\mathbf{u}^2-Q_0^2)r^{3/2}\times\left(\frac{q}{r}+Q_0\right)(1-e^{-1/2r})\right\}. \tag{A12}$$

Thus the equation for r determines its fixed value r^*, and upon substituting in the second equation above, q^* is determined. In fact, from Eq. (A12) one obtains

$$\frac{q^*}{r^*}=Q_0(1+3\mathbf{u}^2-3Q_0^2)/(1+\mathbf{u}^2-Q_0^2). \tag{A13}$$

If λ increases with time with the schedule of Eq. (38) with $\nu=1$, Eq. (40) becomes

$$\frac{dr}{d\alpha}=\frac{f(r)}{\sqrt{\lambda_0\alpha}}+\frac{2r}{\alpha}. \tag{A14}$$

The fixed point r^* of the dynamics is obtained from

$$\frac{f(r^*)}{\sqrt{\lambda_0}}+2r^*=0. \tag{A15}$$

As can be seen from Fig. 4, if λ_0 is greater than upper bound $\lambda_{\max}$, there exists no fixed point. We can obtain $\lambda_{\max}$ by taking derivative of Eq. (A14) with respect to α at $r=r^*$ which yields

$$\lambda_{\max}=\frac{[f'(r^*)]^2}{4}. \tag{A16}$$

If $\lambda_0=\lambda_{\max}$, there is one fixed point. If $\lambda_0<\lambda_{\max}$, there are two fixed points, one is stable and the other is unstable (see Fig. 4).

[1] S. I. Amari, IEEE Trans. Electron. Comput. **16**, 299 (1967).
[2] H. White, J. Am. Stat. Assoc. **84**, 1003 (1989).
[3] G. Radons, H. Schuster, and D. Werner, in *International Neural Network Conference '90 Paris* (Kluwer Academic, Dordrecht, 1990), p. 993.
[4] T. Heskes and B. Kappen, Phys. Rev. A **44**, 2718 (1991).
[5] T. Heskes, E. T. P. Slijpen, and B. Kappen, Phys. Rev. A **46**, 5221 (1992).
[6] L. K. Hansen, R. Pathria, and P. Salamon, J. Phys. A **26**, 63 (1993).
[7] G. Radons, J. Phys. A **26**, 3455 (1993).
[8] T. Heskes, J. Phys. A **27**, 5145 (1994).
[9] M. Biehl and H. Schwarze, J. Phys. A **28**, 643 (1995).
[10] P. Riegler and M. Biehl, J. Phys. A **28**, L507 (1995).
[11] D. Saad and S. Solla, Phys. Rev. Lett. **74**, 4337 (1995).
[12] D. Saad and S. Solla, Phys. Rev. E **52**, 4225 (1995).

[13] C. W. H. Mace and A. C. C. Coolen, Stat. Comput. (to be published).
[14] N. Barkai, H. S. Seung, and H. Sompolinsky, Phys. Rev. Lett. **75**, 1415 (1995).
[15] H. Sompolinsky, N. Barkai, and H. S. Seung, *Neural Networks: The Statistical Mechanics Perspective*, edited by J. Oh, C. Kwon, and S. Cho (World Scientific, Singapore, 1995).
[16] N. Barkai, H. S. Seung, and H. Sompolinsky, in *Advances in Neural Information Systems*, edited by R. P. Lippman, J. E. Moody, and D. S. Touretzky (Kaufmann, San Mateo, CA, 1995), Vol. 7, p. 303.
[17] N. Barkai, Ph.D. thesis, Hebrew University of Jerusalem, 1995.
[18] M. L. Minsky and S. A. Papert, *Perceptrons* (MIT Press, Cambridge, MA, 1969).
[19] M. Biehl, P. Riegler, and M. Stechert, Phys. Rev. E **52**, 4625 (1995).
[20] M. Copelli, O. Kinouchi, and N. Caticha, Phys. Rev. E **53**, 6341 (1996).
[21] S. Agmon, Can. J. Math. **6**, 382 (1954).
[22] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis* (Wiley, New York, 1973).
[23] J. K. Anlauf and M. Biehl, Europhys. Lett. **10**, 687 (1989).
[24] M. Biehl and P. Riegler, Europhys. Lett. **28**, 525 (1994).
[25] H. Sompolinsky and J. W. Kim (unpublished).
[26] H. J. Kushner, SIAM (Soc. Ind. Appl. Math.) J. Appl. Math. **47**, 169 (1987).
[27] O. Kinouchi and N. Caticha, J. Phys. A **25**, 6243 (1992).
[28] M. Copelli and N. Caticha, J. Phys. A **28**, 1615 (1995).
[29] O. Kinouchi and N. Caticha, Phys. Rev. E **53**, 2878 (1995).
[30] M. Copelli, R. Eichhorn, O. Kinouchi, M. Biehl, R. Simonetti, P. Riegler, and N. Caticha, Europhys. Lett. **37**, 432 (1997).
[31] M. Biehl and P. Riegler, Phys. Rev. Lett. **78**, 4305 (1997).
[32] J. W. Kim and H. Sompolinsky, Phys. Rev. Lett. **78**, 4306 (1997).