On-line Gibbs learning. I. General theory

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

We study a model of on-line learning, the on-line Gibbs algorithm (OLGA), which is particularly suitable for supervised learning of realizable and unrealizable discrete valued functions. The learning algorithm is based on an on-line energy function E that balances the need to minimize the instantaneous error against the need to minimize the change in the weights. The relative importance of these terms in E is determined by a parameter λ, the inverse of which plays a similar role as the learning rate parameters in other on-line learning algorithms. In the stochastic version of the algorithm, following each presentation of a labeled example the new weights are chosen from a Gibbs distribution with the on-line energy E and a temperature parameter T. In the zero-temperature version of OLGA, at each step, the new weights are those that minimize the on-line energy E. The generalization performance of OLGA is studied analytically in the limit of small learning rate, i.e., $\lambda \to \infty$. It is shown that at finite temperature OLGA converges to an equilibrium Gibbs distribution of weight vectors with an energy function which is equal to the generalization error function. In its deterministic version OLGA converges to a local minimum of the generalization error. The rate of convergence is studied for the case in which the parameter λ increases as a power law in time. It is shown that in a generic unrealizable dichotomy, the asymptotic rate of decrease of the generalization error is proportional to the inverse square root of presented examples. This rate is similar to that of batch learning. The special cases of learning realizable dichotomies or dichotomies with output noise are also treated. [S1063-651X(98)07808-8]

PACS number(s): 87.10.+e

I. INTRODUCTION

Constructing a general model of on-line learning is an important challenge in the theory of learning and its applications. A plausible definition of the goal of supervised learning from examples is the minimization of the generalization error. In this work we focus mostly on supervised learning of discrete valued functions such as dichotomies, which are extremely useful for decision and classification tasks. A general model of batch learning in which the learner has free access to a fixed set of examples is based on minimization of the total training error. Indeed, as the size of the training set, P, grows this procedure converges uniformly to the minimum of the generalization error. In systems with continuously varying weights, the rate of convergence to this limit follows a power law [1–6]. For example, in learning Boolean functions which can be perfectly realized by the system the generalization error decreases to zero as $1/P$. In learning a generic unrealizable Boolean function, the power law is $1/\sqrt{P}$ [2–12]. A similar general model for on-line learning of discrete valued functions has been lacking. The conventional on-line algorithm is based on the gradient of the instantaneous error [13–26]. For a sufficiently small learning rate, it converges to a local minimum of the generalization error (although not necessarily to the global one). However, this algorithm is not applicable to learning Boolean functions or other discrete valued functions, as their instantaneous error cannot be differentiated. Thus even for the simple single-layer perceptron there has not been an on-line learning algorithm that guarantees convergence to a minimum generalization error for a general unrealizable rule [10,27–29].

The present paper is the first of two papers that investigate in detail a model of on-line learning, which we call the on-line Gibbs algorithm (OLGA). The algorithm is specified by two parameters. One is temperature T which denotes the level of stochasticity in the algorithm. The second parameter denoted as λ characterizes the size of the changes made in each step. The inverse of λ is analogous to the learning rate constant in conventional on-line learning algorithms. OLGA is applicable to learning general functions, including smooth and discrete valued functions. In this paper (paper I) we define the algorithm and study analytically its general properties, using methods of the theory of stochastic processes [15–19,30,31]. In the following paper (paper II) we apply OLGA to specific systems and analyze its behavior using mean-field theory as well as numerical simulations.

The outline of paper I is as follows. In Sec. II we define the algorithm. In Sec. III the general properties of the algorithm at nonzero T are investigated. This is done for the limit of large λ, namely, vanishing learning rate. In Sec. IV the properties of OLGA in the deterministic limit (i.e., $T=0$), and large λ, are derived. In Sec. V the thermodynamic limit of OLGA (in which the number of weights approaches infinity) is investigated. The results are summarized and discussed in Sec. VI.

A preliminary version of the work has been presented in Refs. [32] and [33].

II. THE ON-LINE GIBBS ALGORITHM

A. The algorithm

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, numbered by the integer n, the system is given an input vector $\mathbf{s}^n$ and the desired output $\sigma_0^n = \sigma_0(\mathbf{s}^n)$. The in-

puts 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 $n-1$ steps, 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 which prevents the system from making big changes in the weights at each step. 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.

OLGA is a stochastic update rule. Given the current weights and the new randomly sampled example, the probability that the new weight vector is $\mathbf{w}'$ is given by

$$\mathcal{P}(\mathbf{w}'|\mathbf{w},\mathbf{s})=\frac{1}{Z(\mathbf{w},\mathbf{s})}\exp\left(-\frac{\beta}{2}E(\mathbf{w}'|\mathbf{w},\mathbf{s})\right), \tag{2}$$

which we call *the on-line Gibbs distribution*. The normalization function Z can be written as

$$Z(\mathbf{w},\mathbf{s})=\exp\left(-\frac{\beta}{2}f(\mathbf{w},\mathbf{s})\right), \tag{3}$$

where

$$\frac{\beta}{2}f(\mathbf{w},\mathbf{s})=-\ln\int d\mathbf{w}'\exp\left(-\frac{\beta\lambda}{4}\|\mathbf{w}'-\mathbf{w}\|^2-\frac{\beta}{2}\epsilon(\mathbf{w}',\mathbf{s})\right). \tag{4}$$

Note that f depends on $\mathbf{w}$ and the new example. The full process can be described as a Markov process (for a fixed sequence of examples $\{\mathbf{s}\}$),

$$\mathcal{P}(\mathbf{w}',n;\{\mathbf{s}\})=\int d\mathbf{w}\ \mathcal{P}(\mathbf{w}'|\mathbf{w},\mathbf{s})\mathcal{P}(\mathbf{w},n-1;\{\mathbf{s}\}), \tag{5}$$

where $\mathcal{P}(\mathbf{w},n;\{\mathbf{s}\})$ is the probability density of the weights at time step n which depends on the sequence of n examples that were presented so far.

The algorithm depends on two parameters, T and λ. The temperature T specifies the level of stochasticity in the algorithm. In the limit of zero temperature the algorithm reduces to choosing as the new weight vector one that minimizes E. The parameter λ determines the size of the change in the weights at each presentation.

The free energy, Eq. (4), serves as a generating function for the moments of the change in the weights at each step. Defining

$$\Delta\mathbf{w}=\mathbf{w}'-\mathbf{w}, \tag{6}$$

it is straightforward to see that

$$\langle\Delta\mathbf{w}\rangle=-\frac{1}{\lambda}\frac{\partial}{\partial\mathbf{w}}f(\mathbf{w},\mathbf{s}). \tag{7}$$

The fluctuations in $\Delta\mathbf{w}$ obey

$$\langle\Delta w_i\Delta w_j\rangle_C=\frac{2}{\beta\lambda^2}\left(\frac{-\partial^2}{\partial w_i\partial w_j}f(\mathbf{w},\mathbf{s})+\lambda\delta_{ij}\right), \tag{8}$$

where $\langle\Delta w_i\Delta w_j\rangle_C\equiv\langle\Delta w_i\Delta w_j\rangle-\langle\Delta w_i\rangle\langle\Delta w_j\rangle$ and $\langle\ \rangle$ is a *thermal* average, i.e., average over $\mathbf{w}'$ using the on-line Gibbs distribution $\mathcal{P}(\mathbf{w}'|\mathbf{w},\mathbf{s})$, Eq. (2). In a similar manner, higher-order connected correlations of $\Delta\mathbf{w}$ can be derived by higher-order derivatives of $f(\mathbf{w},\mathbf{s})$.

B. System architecture

We will consider in this work the following classes of network architectures.

Smooth networks. This class is defined by the conditions that the error function $\epsilon(\mathbf{w},\mathbf{s})$ is twice differentiable with respect to $\mathbf{w}$. In the context of multilayered perceptrons it requires that all the neurons are smooth sigmoidal functions of their inputs and that the error function is a smooth function of the output of the system. Of course a network with threshold units does not fall into this class.

Networks with discrete output. In this class the output of the system is of the form

$$\sigma(\mathbf{w},\mathbf{s})=F(\{g_k(\mathbf{w},\mathbf{s})\}). \tag{9}$$

The function F is a function of a set of variables $\{g_k\}$ which has discontinuities at $g_k=0$. The variables $g_k(\mathbf{w},\mathbf{s})$ are assumed to be differentiable functions of the weights. Finally, it is assumed that there is a finite density of inputs at the decision boundaries of the system. These boundaries are defined by the vectors $\mathbf{s}$ that obey $g_k(\mathbf{w},\mathbf{s})=0$.

A simple case is the *threshold-smooth network*, the output of which is of the form

$$\sigma(\mathbf{w},\mathbf{s})=\mathrm{sgn}[g(\mathbf{w},\mathbf{s})], \tag{10}$$

where g is a smooth function of $\mathbf{w}$. An example is the single-layer *perceptron* [34]

$$\sigma(\mathbf{w};\mathbf{s})=\mathrm{sgn}(\mathbf{w}\cdot\mathbf{s}), \tag{11}$$

where both $\mathbf{s}$ and $\mathbf{w}$ are N-component vectors. Here g is a linear function of the weights. Another example of a threshold-smooth network is a multilayer network with a *threshold* output unit and sigmoidal hidden units,

$$\sigma(\{\mathbf{w}_l\},\mathbf{s})=\mathrm{sgn}\left(\sum_{l=1}^{M}J_l\tanh(\mathbf{w}_l\cdot\mathbf{s})\right), \tag{12}$$

where $\mathbf{w}_l$ is the weight vector of the lth hidden unit and J_l is the weight connecting the lth hidden unit to the output unit. Here $g_l(\mathbf{w},\mathbf{s})=\tanh(\mathbf{w}_l\cdot\mathbf{s})$.

Another important case of a threshold-smooth network is a two-layer committee machine [35]. Here

$$\sigma(\{\mathbf{w}_k\},\mathbf{s})=\mathrm{sgn}\left(\sum_{k=1}^{M}\mathrm{sgn}(\mathbf{w}_k\cdot\mathbf{s})\right), \tag{13}$$

where $\mathbf{w}_k$ is the weight vector of the kth perceptron ''committee member.'' Here $g_k(\mathbf{w},\mathbf{s})=\mathrm{sgn}(\mathbf{w}_k\cdot\mathbf{s})$. An additional example is a winner-takes-all (WTA) network which is appropriate for classification tasks. In this network, the output takes integer values $l=1,\ldots,K$, according to

$$\sigma(\{\mathbf{w}_l\},\mathbf{s})=\mathrm{argmax}_l(\mathbf{w}_l\cdot\mathbf{s}). \tag{14}$$

C. OLGA at $T=0$

At $T=0$ our algorithm reduces to finding at each step the global minimum of the on-line energy, Eq. (1), given the current weights and the new example. We now describe the implementation of the zero-temperature algorithm for the two classes of problems defined above.

Smooth networks. In this case, we may attempt to determine $\mathbf{w}'$ by locally minimizing E, Eq. (1), i.e., by solving for $\mathbf{w}'$ the equation

$$\mathbf{w}'=\mathbf{w}-\frac{1}{\lambda}\nabla\epsilon(\mathbf{w}';\mathbf{s}), \tag{15}$$

where ∇ is gradient with respect to $\mathbf{w}'$. In the general nonlinear case and moderate values of λ, solving these equations may be difficult. In addition, E may have more than one local minimum. However, in general we are interested in the limit of large λ, in which case E is expected to have a unique minimum which can be evaluated by an expansion in $1/\lambda$, as will be shown in Sec. IV A.

Networks with discrete output. The novelty of our algorithm lies mainly in learning nonsmooth problems, such as networks with threshold neurons. As an example we will focus on learning Boolean functions where σ, σ_0, and $\epsilon(\mathbf{w},\mathbf{s})$ take on $\{0,1\}$ values. 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 algorithm 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.,

$$\|\mathbf{w}'-\mathbf{w}\|<\sqrt{2/\lambda}. \tag{16}$$

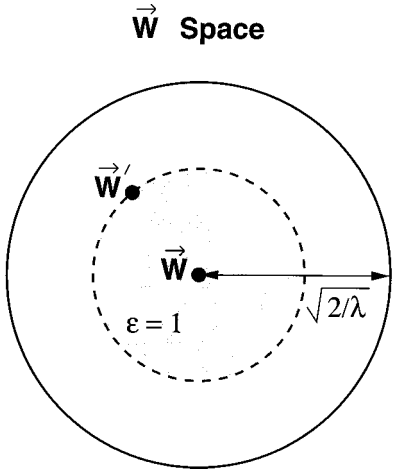

FIG. 1. Schematic illustration of OLGA update rule for a system with binary error. The solid line denotes the hypersphere in weight space centered on the current weight vector $\mathbf{w}$ with a radius $\sqrt{2/\lambda}$. Only if the nearest weight vector $\mathbf{w}'$ which satisfies the new example lies within this hypersphere is an update ($\mathbf{w}\rightarrow\mathbf{w}'$) made. The dashed line denotes the hypersphere on which $\mathbf{w}'$ lies.

Otherwise, $\mathbf{w}'=\mathbf{w}$. See Fig. 1. In general, this computation may be intensive. However, as we will demonstrate later, in many cases one can write explicitly relatively simple update rules that implement the above algorithm. First, we derive some general properties of the algorithm.

D. Generalization error

As in other on-line supervised learning algorithms the performance of OLGA is measured by the resultant generalization error. The generalization function $\epsilon_g(\mathbf{w})$ is defined as

$$\epsilon_g(\mathbf{w})=\langle\langle\epsilon(\mathbf{w};\mathbf{s})\rangle\rangle, \tag{17}$$

where $\langle\langle\ \rangle\rangle$ denotes average with respect to the input distribution $D\mathbf{s}$. To measure the average generalization error of the learning algorithm we define the probability distribution

$$\mathcal{P}(\mathbf{w},n)=\langle\langle\mathcal{P}(\mathbf{w},n;\{\mathbf{s}\})\rangle\rangle, \tag{18}$$

where the average is over all possible sequences of n examples. The equilibrium weight distribution is

$$\mathcal{P}_\infty(\mathbf{w})=\lim_{n\rightarrow\infty}\mathcal{P}(\mathbf{w},n). \tag{19}$$

The (average) generalization error at step n is defined as

$$\epsilon_g(n)=\int d\mathbf{w}\mathcal{P}(\mathbf{w},n)\epsilon_g(\mathbf{w}). \tag{20}$$

Finally, the equilibrium generalization error is defined as the long time limit of the average generalization error

$$\epsilon_\infty=\lim_{n\rightarrow\infty}\epsilon_g(n)=\int d\mathbf{w}\mathcal{P}_\infty(\mathbf{w})\epsilon_g(\mathbf{w}). \tag{21}$$

III. GENERAL RESULTS FOR LARGE λ AT NONZERO *T*

In this section we analyze the general properties of OLGA in the limit $\lambda\to\infty$ while the temperature is kept at a fixed positive value. In Appendix A we show that in this limit the leading terms in the on-line free energy, Eq. (4), are of the form

$$f(\mathbf{w},\mathbf{s})=\frac{N}{\beta}\ln\left(\frac{\lambda\beta}{4\pi}\right)+\epsilon(\mathbf{w},\mathbf{s})+\delta(\mathbf{w},\mathbf{s}), \tag{22}$$

where $\delta(\mathbf{w},\mathbf{s})$ vanishes in the large λ limit. Substituting Eq. (22) in Eqs. (7) and (8) and averaging over **s**, we obtain

$$\langle\langle\langle\Delta\mathbf{w}\rangle\rangle\rangle=-\frac{1}{\lambda}\nabla\epsilon_g(\mathbf{w})+O(\lambda^{-1-\nu}), \tag{23}$$

$$\langle\langle\langle\Delta w_i\Delta w_j\rangle\rangle\rangle=\frac{2}{\beta\lambda}\delta_{ij}+O(\lambda^{-1-\nu}), \tag{24}$$

where $\nu=1$ in the smooth case and $\nu=\frac{1}{2}$ in the threshold-smooth case. See Appendix A.

Equations (23) and (24) are our fundamental results for the behavior of the algorithm at large λ and finite *T*. These equations imply that the weight vector executes a random walk with a step size of the order of $1/\sqrt{\beta\lambda}$. This thermal noise is mostly due to the isotropic quadratic term in *E*. The mean drift of the weight vector is of the order of $1/\lambda$. It consists of a gradient descent with the generalization error as the potential. These results can be cast in the following stochastic equations:

$$\Delta\mathbf{w}^n=-\frac{1}{\lambda}\nabla\epsilon_g(\mathbf{w})+\mathbf{z}^n, \tag{25}$$

where $\mathbf{z}^n$ is Gaussian uncorrelated noise, with zero mean and

$$\langle z_i^n z_j^m\rangle=\frac{2T}{\lambda}\delta_{nm}\delta_{ij}. \tag{26}$$

We now consider the temporal evolution of the *average* weight distribution, Eq. (18). This distribution can also be written as

$$\mathcal{P}(\mathbf{w},n)=\langle\langle\delta(\mathbf{w}-\mathbf{w}^{n-1}-\Delta\mathbf{w}^n)\rangle_{\Delta\mathbf{w}}\rangle_{n-1}, \tag{27}$$

where $\langle\ \rangle_{\Delta\mathbf{w}}$ denotes thermal and quenched average over the increments at the *n* step, i.e., integrating with the distribution $\mathcal{P}(\mathbf{w}+\Delta\mathbf{w}^n|\mathbf{w})=\langle\langle\mathcal{P}(\mathbf{w}+\Delta\mathbf{w}^n|\mathbf{w},\mathbf{s})\rangle\rangle$. The average $\langle\cdots\rangle_{n-1}$ denotes average over all possible weight vectors at the $n-1$ step, i.e., integrating with the distribution $\mathcal{P}(\mathbf{w}^{n-1},n-1)=\langle\langle\mathcal{P}(\mathbf{w}^{n-1},n-1;\{\mathbf{s}\})\rangle\rangle$. Expanding the right hand side of the above equation in powers of $\Delta\mathbf{w}^n$ and using the results of the preceding paragraph, we obtain

$$\Delta\mathcal{P}(\mathbf{w},n)\equiv\mathcal{P}(\mathbf{w},n)-\mathcal{P}(\mathbf{w},n-1) \tag{28}$$

$$=\frac{2}{\beta\lambda}\left\langle\left(\frac{1}{2}\beta(\nabla\epsilon_g(\mathbf{w}^{n-1}))\cdot\nabla+\frac{1}{2}\nabla^2\right)\delta(\mathbf{w}-\mathbf{w}^{n-1})\right\rangle_{n-1}+O(\lambda^{-1-\nu}). \tag{29}$$

A proper large λ limit is obtained by defining a scaled time variable

$$\tau=n/\lambda \tag{30}$$

and constructing the continuous-time limit by

$$\mathcal{P}(\mathbf{w},\tau)=\lim_{\lambda\to\infty}\mathcal{P}(\mathbf{w},\lambda\tau). \tag{31}$$

Using the above result we obtain that $\mathcal{P}(\mathbf{w},\tau)$ obeys the following Fokker-Planck equation [15,19,31]:

$$\frac{\partial}{\partial\tau}\mathcal{P}(\mathbf{w},\tau)=\nabla\cdot[\nabla\epsilon_g(\mathbf{w})\mathcal{P}(\mathbf{w},\tau)]+T\nabla^2\mathcal{P}(\mathbf{w},\tau). \tag{32}$$

This result implies in particular that the stationary distribution of the weights is

$$\mathcal{P}_\infty(\mathbf{w})=Z^{-1}\exp[-\epsilon_g(\mathbf{w})/T]. \tag{33}$$

Thus in the large λ limit at finite *T* our algorithm converges to an equilibrium state which is a Gibbs distribution with $\epsilon_g(\mathbf{w})$ as the energy. This is analogous to the stationary state of the statistical mechanical batch learning in the *high-temperature limit* [1,2].

Our result implies that OLGA converges at sufficiently long time to a distribution which becomes sharply centered at the desired weight vector, i.e., the global minimum of ϵ_g, *in the limit that* $T\to0$ *after taking the limit* $\lambda\to\infty$. To actually reach this target, one has to specify an appropriate schedule for making λ and $1/T$ sufficiently large. This schedule will determine the *rate* of convergence of the system to the optimal performance. In general, to guarantee convergence to the global minimum *T* has to be changed in a slow logarithmic schedule

$$T(n)=\frac{T_0}{\ln(n)}. \tag{34}$$

In many cases, power-law rates are sufficient. We will not discuss further the issue of scheduling λ and *T* for the stochastic OLGA.

IV. RESULTS FOR LARGE λ AT $T=0$

A. Smooth networks

The large λ limit of OLGA in the case of smooth networks can be derived by expanding Eq. (15) in powers of $\mathbf{w}'-\mathbf{w}$. The leading order can be evaluated by substituting **w** in the argument of ϵ, yielding

$$\mathbf{w}' = \mathbf{w} - \frac{1}{\lambda}\nabla\epsilon(\mathbf{w};\mathbf{s}). \tag{35}$$

Thus for smooth problems and large λ our algorithm reduces at zero temperature to the conventional stochastic gradient descent algorithm with a small learning rate, $1/\lambda$. Viewing the stochastic gradient descent algorithm as minimizing an instantaneous cost function has been proposed by Kivinen and Warmuth [36] for linear systems.

From Eq. (35) the low-order statistics of the single step can be obtained,

$$\langle\langle\langle \Delta\mathbf{w}\rangle\rangle\rangle = -\frac{1}{\lambda}\nabla\epsilon_g(\mathbf{w}) + O(\lambda^{-2}), \tag{36}$$

$$\langle\langle\langle \Delta w_i \Delta w_j\rangle\rangle\rangle = \frac{2}{\lambda^2}T_{ij}(\mathbf{w}) + O(\lambda^{-3}). \tag{37}$$

The *diffusion matrix* is

$$T_{ij}(\mathbf{w}) = \tfrac{1}{2}\langle\langle \partial_i\epsilon(\mathbf{w},\mathbf{s})\partial_j\epsilon(\mathbf{w},\mathbf{s})\rangle\rangle, \tag{38}$$

where $\partial_i \equiv \partial/\partial w_i$. The higher-order moments are of a magnitude that is higher order in $1/\lambda$ and can be neglected.

Using methods similar to that of the preceding section one can define a scaled time variable

$$\tau = n/\lambda \tag{39}$$

and derive a continuous time Fokker-Planck equation for the weight distribution,

$$\frac{\partial}{\partial\tau}\mathcal{P}(\mathbf{w},\tau) = \nabla\cdot[\nabla\epsilon_g(\mathbf{w})\mathcal{P}(\mathbf{w},\tau)] + \frac{1}{\lambda}\sum_{ij}\partial_i\partial_j[T_{ij}(\mathbf{w})\mathcal{P}(\mathbf{w},\tau)]. \tag{40}$$

These results are similar to the standard on-line gradient descent algorithm in the limit of zero learning rate. The above equation implies that as long as $\nabla\epsilon_g(\mathbf{w})$ is not small, the diffusion term which is of the order of $1/\lambda$ can be neglected and the evolution can be described as a deterministic gradient descent towards a local minimum of $\epsilon_g(\mathbf{w})$, to be denoted by $\mathbf{w}^*$. Near the local minimum the drifting ''force'' is small and is comparable to the diffusion term. To describe the dynamics near the local minimum we write

$$\mathbf{u} = (\mathbf{w} - \mathbf{w}^*)\sqrt{\lambda}. \tag{41}$$

The distribution of these scaled weights obeys

$$\frac{\partial}{\partial\tau}\mathcal{P}(\mathbf{u},\tau) = H_{ij}\partial_i[u_j\mathcal{P}(\mathbf{u},\tau)] + \sum_{ij}T_{ij}\partial_i\partial_j\mathcal{P}(\mathbf{u},\tau). \tag{42}$$

The matrix H_{ij} is the Hessian of $\epsilon_g(\mathbf{w})$, namely, $H_{ij} = \partial_i\partial_j\epsilon_g(\mathbf{w}^*)$. The matrix T_{ij} is the diffusion matrix evaluated at $\mathbf{w}^*$. Thus the equilibrium distribution of $\mathbf{w}$ has a form of a Gaussian peaked at the local minimum of ϵ_g, i.e., at $\mathbf{w}^*$, with a width of the order $1/\sqrt{\lambda}$. Finally, the equilibrium generalization error is

$$\epsilon_\infty = \epsilon_{\min} + \frac{1}{2\lambda}\sum_{ij}H_{ij}\langle u_i u_j\rangle, \tag{43}$$

where the average over $\mathbf{u}$ is with respect to the equilibrium distribution $\mathcal{P}_\infty(\mathbf{u})$.

B. Generic threshold-smooth networks

Due to the threshold operation of the network's output the large λ limit of the deterministic algorithm is more complicated than the previous case. For fixed input $\mathbf{s}$, if $\sigma_0 g(\mathbf{w},\mathbf{s}) < 0$, the minimal change in weight $\Delta\mathbf{w} = \mathbf{w}' - \mathbf{w}$ that will correct the error is such that

$$0 = g(\mathbf{w}',\mathbf{s}) \approx g(\mathbf{w},\mathbf{s}) + \Delta\mathbf{w}\cdot\nabla g(\mathbf{w},\mathbf{s}), \tag{44}$$

where we have kept only the first-order terms in $\|\Delta\mathbf{w}\|$ since the magnitude of the step will be small in the large λ limit, and ∇ stands for gradient with respect to $\mathbf{w}$ unless stated otherwise. Minimizing with respect to $\Delta\mathbf{w}$ yields

$$\Delta\mathbf{w} = -g(\mathbf{w},\mathbf{s})\nabla g(\mathbf{w},\mathbf{s})/\|\nabla g(\mathbf{w},\mathbf{s})\|^2. \tag{45}$$

Incorporating the bounded change condition, the above update is executed if and only if

$$0 < -\sigma_0 g(\mathbf{w},\mathbf{s})/\|\nabla g(\mathbf{w},\mathbf{s})\| < \sqrt{2/\lambda}. \tag{46}$$

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

It is convenient to introduce the notation

$$\tilde{g}(\mathbf{w},\mathbf{s}) \equiv g(\mathbf{w},\mathbf{s})/\|\nabla g(\mathbf{w},\mathbf{s})\|. \tag{47}$$

Note that in the large λ limit whenever there is a change in the weights,

$$\nabla\tilde{g} \approx \nabla g(\mathbf{w},\mathbf{s})/\|\nabla g(\mathbf{w},\mathbf{s})\| \tag{48}$$

since a change is made only if $g(\mathbf{w},\mathbf{s}) \approx 0$. Thus the update rule in the large λ limit is

$$\Delta\mathbf{w} \approx -\tilde{g}(\mathbf{w},\mathbf{s})\nabla\tilde{g}(\mathbf{w},\mathbf{s}) \tag{49}$$

provided that

$$0 < -\sigma_0\tilde{g}(\mathbf{w},\mathbf{s}) < \sqrt{2/\lambda}, \tag{50}$$

see Fig. 2. Therefore the average of $\Delta\mathbf{w}$ in a single step is

$$\langle\langle\Delta\mathbf{w}\rangle\rangle = \lim_{\lambda\to\infty}\langle\langle -\tilde{g}(\mathbf{w},\mathbf{s})\nabla\tilde{g}(\mathbf{w},\mathbf{s})\Theta(-\sigma_0\tilde{g}(\mathbf{w},\mathbf{s}))\Theta(\sigma_0\tilde{g}(\mathbf{w},\mathbf{s}) + \sqrt{2/\lambda})\rangle\rangle, \tag{51}$$

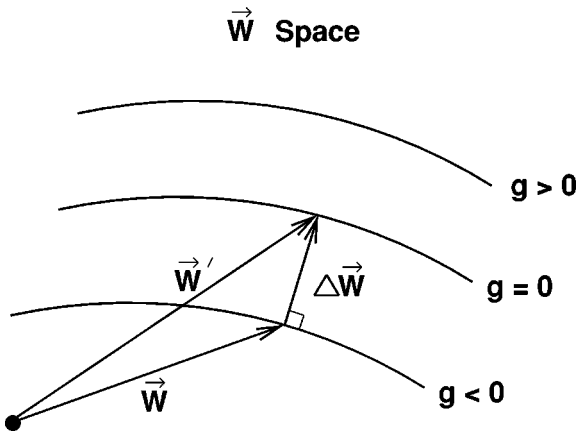

FIG. 2. Schematic illustration of OLGA in the case of large λ. Solid lines show surfaces in weight space defined by constant values of $g(\mathbf{w},\mathbf{s})$. The vector $\mathbf{s}$ is the input of the new example. When an update is made the new weight vector is $\mathbf{w}'=\mathbf{w}+\Delta\mathbf{w}$ where $\Delta\mathbf{w}$ is orthogonal to the surface $g(\mathbf{w},\mathbf{s})$ and its magnitude is such that $\mathbf{w}'$ lies on the surface $g(\mathbf{w}',\mathbf{s})=0$, see Eq. (49).

where as before $\langle\langle\ \ \rangle\rangle$ is the average over input distribution. Taking into account only these inputs and assuming that the input distribution $P_0(\mathbf{s})$ is smooth and nonvanishing at decision boundary, we decompose $\mathbf{s}$ as $\mathbf{s}=\mathbf{s}_\perp+\mathbf{s}_\parallel$ where $g(\mathbf{w},\mathbf{s}_\perp)=0$ and $\mathbf{s}_\parallel$ is orthogonal to the decision boundary and decompose the average over $\mathbf{s}$ as an average over $\mathbf{s}_\parallel$ and $\mathbf{s}_\perp$, separately. Since $\|\Delta\mathbf{w}\|$ for fixed $\mathbf{w}$ is $O(1/\sqrt{\lambda})$, the inputs which satisfy the bounded change condition lie in the narrow band which is $O(1/\sqrt{\lambda})$ from the decision boundary $g(\mathbf{w},\mathbf{s})=0$. See Fig. 3. Thus

$$\langle\langle\Delta\mathbf{w}\rangle\rangle\approx\int_0^{\sqrt{2/\lambda}}dzz\langle\langle\sigma_0(\mathbf{s}_\perp)\nabla\tilde{g}(\mathbf{w},\mathbf{s}_\perp)\rangle\rangle_{P_0(\mathbf{s}_\perp)} \tag{52}$$

from which we obtain

$$\langle\langle\Delta\mathbf{w}\rangle\rangle\approx\frac{1}{\lambda}\langle\langle\sigma_0\nabla\tilde{g}(\mathbf{w},\mathbf{s}_\perp)\rangle\rangle_{P_0(\mathbf{s}_\perp)} \tag{53}$$

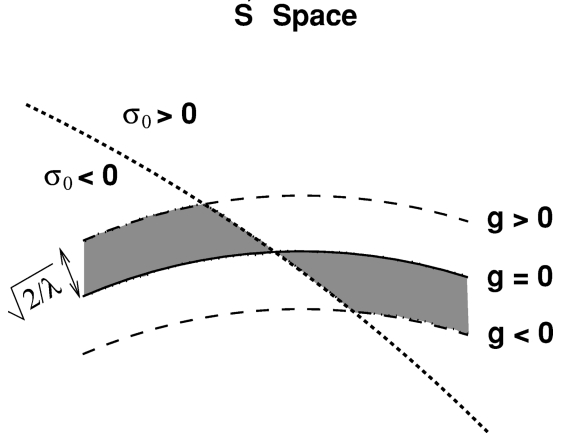

FIG. 3. Average of $\Delta\mathbf{w}$ over the input space for fixed $\mathbf{w}$. Solid line and dotted line show the decision boundaries of the student $[g(\mathbf{w},\mathbf{s})=0]$, and the teacher $[g(\mathbf{w}_0,\mathbf{s})=0]$, respectively. The shaded regions denote the inputs that induce changes in the weights, thus contributing to the average of $\Delta\mathbf{w}$, see Eqs. (51) and (52).

$$=\frac{1}{\lambda}\langle\langle\sigma_0\delta(g(\mathbf{w},\mathbf{s}))\nabla g(\mathbf{w},\mathbf{s})\rangle\rangle. \tag{54}$$

Since

$$\nabla\epsilon_g(\mathbf{w})=\nabla\langle\langle\Theta(-\sigma_0 g(\mathbf{w},\mathbf{s}))\rangle\rangle \tag{55}$$

$$=\langle\langle\delta(g(\mathbf{w},\mathbf{s}))[-\sigma_0\nabla g(\mathbf{w},\mathbf{s})]\rangle\rangle, \tag{56}$$

we obtain

$$\langle\langle\Delta\mathbf{w}\rangle\rangle=-\frac{1}{\lambda}\nabla\epsilon_g(\mathbf{w}) \tag{57}$$

in the large λ limit. In a similar fashion we obtain

$$\langle\langle\Delta w_i\Delta w_j\rangle\rangle=\frac{2}{\lambda^{3/2}}T_{ij}(\mathbf{w}), \tag{58}$$

where

$$T_{ij}(\mathbf{w})=\frac{\sqrt{2}}{3}\langle\langle\delta(g(\mathbf{w},\mathbf{s}))\partial_i g(\mathbf{w},\mathbf{s})\partial_j g(\mathbf{w},\mathbf{s})/\|\nabla g(\mathbf{w},\mathbf{s})\|\rangle\rangle. \tag{59}$$

Finally, it can be shown that the leading correction to Eq. (57) is of the order $O(1/\lambda^{3/2})$. These results imply that the distribution of $\mathbf{w}$ obeys a Fokker-Planck equation in terms of scaled time

$$\tau=n/\lambda, \tag{60}$$

$$\frac{\partial}{\partial\tau}\mathcal{P}(\mathbf{w},\tau)=\nabla\cdot[\nabla\epsilon_g(\mathbf{w})\mathcal{P}(\mathbf{w},\tau)]+\frac{1}{\sqrt{\lambda}}\partial_i\partial_j[T_{ij}(\mathbf{w})\mathcal{P}(\mathbf{w},\tau)]. \tag{61}$$

Thus, near a local minimum, the proper scale of the fluctuations of the weights is

$$\mathbf{u}=(\mathbf{w}-\mathbf{w}^*)\lambda^{1/4} \tag{62}$$

and the distribution of $\mathbf{u}$ obeys a Fokker-Planck equation of the form of Eq. (42). In particular the equilibrium distribution of $\mathbf{w}$ is Gaussian with a mean $\mathbf{w}^*$ and a width of the order $\|\delta\mathbf{w}\|$ of the order $1/\lambda^{1/4}$. Thus

$$\epsilon_\infty\approx\epsilon_{\min}+O(\delta\mathbf{w})^2\approx\epsilon_{\min}+O\left(\frac{1}{\sqrt{\lambda}}\right). \tag{63}$$

C. Threshold-smooth networks with output noise

In this section we consider the case of a threshold network that learns a rule generated by a teacher of the same architecture as the learning system except for an output noise in the teacher's labels. Thus the labels provided by the teacher are of the form

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

where $0\leqslant p<1/2$ is the probability of error in the label provided by the teacher. This form of noise is special in that it

generates errors with a probability which is independent of the inputs. Other than this noise the rule is realizable by the student. Although Eq. (51) holds also for this case, the evaluation of the large λ limit in this equation which was performed in the preceding paragraph is not valid here. The reason for this is the fact that the density of inputs that generate errors (other than those generated by the output noise) is itself vanishing as $\mathbf{w}\to\mathbf{w}_0$. Therefore for $\mathbf{w}$ close to $\mathbf{w}_0$ both σ_0 and g change signs for the same inputs. Hence, integrals over the decision boundary, such as in Eqs. (52) and (54), are meaningless at $\mathbf{w}=\mathbf{w}_0$.

In Appendix B we outline the derivation of the large λ limit in the case of output noise near the global minimum $\mathbf{w}=\mathbf{w}_0$. We find that if we scale the weights as

$$\mathbf{u}=(\mathbf{w}-\mathbf{w}_0)\sqrt{\lambda}, \tag{64}$$

then the equations of the moments of $\mathbf{u}$ are of the form

$$\langle\langle \Delta u_i\rangle\rangle = -\frac{1}{\sqrt{\lambda}}F_i(\mathbf{u}), \tag{65}$$

$$\langle\langle \Delta u_i \Delta u_j\rangle\rangle = \frac{2}{\sqrt{\lambda}}T_{ij}(\mathbf{u}) \tag{66}$$

[see Appendix B for expressions for $F_i(\mathbf{u})$ and $T_{ij}(\mathbf{u})$]. Similar equations (with the same prefactor $1/\sqrt{\lambda}$) hold for higher-order moments. This form implies that one can write down a continuous-time master equation for $\mathcal{P}(\mathbf{u},\tau)$,

$$\frac{\partial}{\partial\tau}\mathcal{P}(\mathbf{u},\tau)=\int D\mathbf{u}'\,T(\mathbf{u}'|\mathbf{u})\mathcal{P}(\mathbf{u}',\tau), \tag{67}$$

with

$$\tau=\frac{n}{\sqrt{\lambda}} \tag{68}$$

and $T(\mathbf{u}'|\mathbf{u})$ is a time-independent transition matrix. This master equation does not have a λ dependence. It converges in general to a non-Gaussian equilibrium distribution $\mathcal{P}_\infty(\mathbf{u})$. Thus the fluctuations of $\mathbf{w}$ about $\mathbf{w}_0$ are of the order

$$\|\mathbf{w}-\mathbf{w}_0\|=O\left(\frac{1}{\sqrt{\lambda}}\right) \tag{69}$$

and

$$\epsilon_\infty\approx\epsilon_{\min}+O(\|\delta\mathbf{w}\|)=p+O\left(\frac{1}{\sqrt{\lambda}}\right) \tag{70}$$

[see Eq. (91) below].

D. Time-dependent λ

Until now we considered the case of fixed and large λ and derived the dependence of the equilibrium average generalization error on λ. Now we consider the *time dependence* of the generalization error in the case where λ is chosen to grow with time. We will consider here only the case of a fixed power-law schedule for λ,

$$\lambda(n)=\lambda_0 n^\nu. \tag{71}$$

The effect of such a schedule has been discussed in detail [10,27–29], in the context of time-dependent learning rates of on-line learning. Essentially, the continuous-time master (or Fokker-Planck) equation for $\mathcal{P}(\mathbf{w},\tau)$ for fixed large λ can be mapped into the corresponding equation for the case of time-dependent λ by substituting $\lambda(\tau)$ for λ provided λ does not increase too fast. The maximum allowed rate of increase is such that the scaled time τ is constant. We now summarize briefly the results for the power-law schedule of λ in the cases analyzed in the previous paragraphs.

(i) Smooth networks. According to the scaling of time, Eq. (39), the power law must obey

$$0\leqslant\nu\leqslant 1. \tag{72}$$

In the case of less than optimal schedule, i.e., $\nu<1$, there are no restrictions on the prefactor λ_0, and the scaling of the fluctuations in the weights with time is

$$\mathbf{w}-\mathbf{w}^*=O\left(\frac{1}{\sqrt{n^\nu}}\right). \tag{73}$$

The average generalization error approaches $\epsilon_{\min}$ as

$$\epsilon_g(n)\approx\epsilon_{\min}+O\left(\frac{1}{n^\nu}\right), \tag{74}$$

see Eqs. (41) and (43).

The optimal schedule is

$$\lambda(n)=\lambda_0 n, \tag{75}$$

where λ_0 must be less than a system-dependent λ_C. In this case

$$\mathbf{w}-\mathbf{w}^*=O\left(\frac{1}{\sqrt{n}}\right), \tag{76}$$

$$\epsilon_g(n)\approx\epsilon_{\min}+O\left(\frac{1}{n}\right). \tag{77}$$

(ii) Generic threshold-smooth networks. According to Eq. (60) the scaling of time with λ is, in this case, the same as in the smooth networks. Hence, the allowed range of ν is the same as Eq. (72). According to Eq. (62), the scale of the fluctuations in the weights is different,

$$\mathbf{w}-\mathbf{w}^*=O\left(\frac{1}{n^{\nu/4}}\right) \tag{78}$$

and similarly,

$$\epsilon_g(n)\approx\epsilon_{\min}+O\left(\frac{1}{\sqrt{n^\nu}}\right). \tag{79}$$

The optimal behavior is

$$\lambda(n)=\lambda_0 n, \tag{80}$$

$$\mathbf{w}-\mathbf{w}^{*}=O\left(\frac{1}{n^{1/4}}\right), \tag{81}$$

$$\epsilon_g(n)\approx\epsilon_{\min}+O\left(\frac{1}{\sqrt{n}}\right). \tag{82}$$

(iii) Threshold network with output noise. In this case the scaling of time is different, as seen from Eq. (68), and the corresponding range of ν is

$$0\leqslant\nu\leqslant 2. \tag{83}$$

On the other hand, the scale of the fluctuations is the same as that of smooth networks, as seen by Eq. (64). If the power law is less than optimal, i.e., $\nu<2$, then for all positive values of the coefficient λ_0 the generalization error scales as

$$\epsilon_g(n)\approx\epsilon_{\min}+O\left(\frac{1}{\sqrt{n^{\nu}}}\right). \tag{84}$$

The optimal behavior is obtained for

$$\lambda=\lambda_0 n^2,\quad \lambda_0<\lambda_C \tag{85}$$

in which case

$$\epsilon_g(n)\approx\epsilon_{\min}+O\left(\frac{1}{n}\right). \tag{86}$$

For optimal schedules of λ, λ_0 has to be smaller than a system-dependent cutoff λ_C. Since λ_C depends on unknown target rule and input distribution in general, the upper bound of λ_C is not known to the learner. If $\lambda_0>\lambda_C$, then ϵ_g converges to $\epsilon_{\min}$ with suboptimal rate in the generic case and ϵ_g remains finite in the pure output noise case [33].

E. Realizable rules at $T=0$ and $\lambda=0$

The problem of learning a realizable rule in a threshold network is a special case of the preceding section with zero noise ($p=0$). The main difference between the noisy case and the noiseless case is that in the case of a realizable rule OLGA converges asymptotically to the teacher weight vector even for small values of λ. The reason for this is that if $\mathbf{w}$ is close to the teacher vector, $\mathbf{w}_0$, then the minimal change that corrects the instantaneous error is small. This is because when $\mathbf{w}$ is equal to $\mathbf{w}_0$ all examples are satisfied. Thus, even without the bounded-change condition imposed by large λ the instantaneous change in the weights is small due to the minimal change condition and becomes vanishingly small as $\mathbf{w}$ approaches $\mathbf{w}_0$. On the other hand, when $\mathbf{w}$ is far from $\mathbf{w}_0$ the algorithm will keep generating a random walk in the $\mathbf{w}$ space, ensuring that eventually the system will visit the neighborhood of $\mathbf{w}_0$ and will be attracted to it. To show this explicitly, and to evaluate the asymptotic convergence rate we consider the simplest case $\lambda=0$. Let us assume that

$$\mathbf{u}\equiv\mathbf{w}-\mathbf{w}_0 \tag{87}$$

is small in magnitude and use Eq. (51) to evaluate the average of the change in $\mathbf{u}$,

$$\langle\langle\Delta\mathbf{u}\rangle\rangle \approx\langle\langle-\tilde{g}(\mathbf{w}_0+\mathbf{u},\mathbf{s})\nabla\tilde{g}(\mathbf{w}_0+\mathbf{u},\mathbf{s})\Theta(-\sigma_0\tilde{g}(\mathbf{w}_0+\mathbf{u},\mathbf{s}))\rangle\rangle. \tag{88}$$

Since $\sigma_0\tilde{g}(\mathbf{w}_0,\mathbf{s})$ is positive for all $\mathbf{s}$, nonzero contribution to $\Delta\mathbf{u}$ comes only from inputs near the decision boundary of $\mathbf{w}_0$, namely, for those inputs for which $\tilde{g}(\mathbf{w}_0,\mathbf{s})$ is of the order of $\nabla\tilde{g}\cdot\mathbf{u}$. Integrating over this region in the input space we obtain in the limit of small $\|\mathbf{u}\|$,

$$\langle\langle\Delta\mathbf{u}\rangle\rangle\approx-\left\langle\!\!\left\langle\delta(g)\nabla g\frac{(\nabla g\cdot\mathbf{u})^2}{\|\nabla g\|^2}\right\rangle\!\!\right\rangle, \tag{89}$$

where g is evaluated at $\mathbf{w}_0$. In this equation in the large n limit, the solution of this difference equation is of the form

$$\mathbf{u}(n)\approx\frac{\mathbf{u}_0}{n}. \tag{90}$$

Finally, in the realizable case ϵ_g is of the form

$$\epsilon_g(\mathbf{w}_0+\mathbf{u})\approx\langle\langle\delta(g)|\nabla g\cdot\mathbf{u}|\rangle\rangle, \tag{91}$$

from which we conclude that

$$\epsilon_g(n)=O\left(\frac{1}{n}\right). \tag{92}$$

V. THE THERMODYNAMIC LIMIT

Another interesting limit where the behavior of the algorithm can be analyzed is the case where the number of weights N is large, in which case an appropriate thermodynamic limit can be defined. We will use scales of weights such that the components w_i are of order unity (i.e., their magnitude remains finite as N goes to ∞). In addition, the error function $\epsilon(\mathbf{w},\mathbf{s})$ is of order 1. A nontrivial thermodynamic limit requires that

$$\beta=\frac{N}{\tilde{T}}, \tag{93}$$

where $\tilde{T}$ is of order 1. This guarantees that the on-line free energy is extensive.

It can be shown that in the large N limit the statistics of $\Delta\mathbf{w}$ is again Gaussian with the following moments:

$$\langle\langle\ \langle\Delta\mathbf{w}\rangle\ \rangle\rangle=-\frac{1}{\lambda}\nabla\tilde{\epsilon}(\mathbf{w}), \tag{94}$$

$$\langle\langle\ \langle\Delta w_i\Delta w_j\rangle\ \rangle\rangle=\frac{1}{N\lambda}T_{ij}(\mathbf{w}), \tag{95}$$

where

$$\tilde{\epsilon}(\mathbf{w})=\langle\langle\tilde{\epsilon}(\mathbf{w},\mathbf{s})\rangle\rangle \tag{96}$$

and

$$T_{ij}(\mathbf{w})=2\tilde{T}\delta_{ij}+\frac{N}{\lambda}\langle\langle\nabla_i\tilde{\epsilon}(\mathbf{w},\mathbf{s})\nabla_j\tilde{\epsilon}(\mathbf{w},\mathbf{s})\rangle\rangle, \tag{97}$$

where as before $\langle\ \rangle$ and $\langle\langle\ \rangle\rangle$ denote thermal average and average over the examples, respectively. The function $\tilde{\epsilon}(\mathbf{w},\mathbf{s})$ is an effective instantaneous error function. For smooth networks it is given by

$$\tilde{\epsilon}(\mathbf{w},\mathbf{s})=\epsilon(\mathbf{w},\mathbf{s})-\frac{1}{\lambda}\|\nabla\epsilon(\mathbf{w},\mathbf{s})\|^2. \tag{98}$$

For thresholded smooth networks, Eq. (10), it is

$$\tilde{\epsilon}(\mathbf{w},\mathbf{s})=\epsilon(\mathbf{w},\mathbf{s})[\Theta(\tfrac{1}{2}\lambda\tilde{g}(\mathbf{w},\mathbf{s})^2-1)+\Theta(1-\tfrac{1}{2}\lambda\tilde{g}^2(\mathbf{w},\mathbf{s}))\tfrac{1}{2}\lambda\tilde{g}^2(\mathbf{w},\mathbf{s})], \tag{99}$$

where $\tilde{g}(\mathbf{w},\mathbf{s})$ is defined in Eq. (A7). Note that in the limit of large λ, $\tilde{\epsilon}(\mathbf{w},\mathbf{s})=\epsilon(\mathbf{w},\mathbf{s})$ and $T_{ij}=2\tilde{T}\delta_{ij}$, and Eqs. (94) and (95) reduce to Eqs. (23) and (24) (with $T\to\tilde{T}N$), as expected. Comparing Eqs. (94) and (95) with Eqs. (23) and (24), it is seen that for large N, keeping λ finite has two effects. First it modifies the effective potential which determines the deterministic part of the dynamics, Eq. (94). Secondly, it modifies the noise term, Eq. (97).

Given the above results, one can derive a continuous-time Fokker-Planck equation for the system, by scaling time as

$$\alpha=n/N. \tag{100}$$

Similar to the derivation of the Fokker-Planck equation in the preceding section, one obtains here

$$\frac{\partial}{\partial\alpha}\mathcal{P}(\mathbf{w},\alpha)=\frac{1}{\lambda}\left\{\nabla\cdot[\nabla\tilde{\epsilon}_g(\mathbf{w})\mathcal{P}(\mathbf{w},\alpha)]+\frac{1}{2}\sum_{ij}\partial_i\partial_j[T_{ij}(\mathbf{w})\mathcal{P}(\mathbf{w},\alpha)]\right\}. \tag{101}$$

Unfortunately, the second contribution to T_{ij} depends on $\mathbf{w}$ and is anisotropic. Therefore when λ is finite there is no simple expression for the equilibrium distribution even for large N. Nevertheless, for specific models, the above averaged dynamics can be reduced to deterministic dynamic equations for a few order parameters which can be solved exactly, as will be demonstrated in paper II.

VI. SUMMARY AND DISCUSSION

In this paper we derived the asymptotic convergence properties of OLGA for fixed λ in the limit of $\lambda\to\infty$ and in the case of a power-law schedule for λ. We summarize here the results for the optimal power-law schedules in the different classes of networks in the deterministic limit of OLGA.

(1) Smooth networks. In the case of smooth systems the optimal behavior is

$$\epsilon_g(n)-\epsilon_{\min}=\frac{A}{n},\quad n\to\infty \tag{102}$$

which is achieved by increasing λ as

$$\lambda(n)=\lambda_0 n. \tag{103}$$

(2) Generic threshold-smooth networks. For general unrealizable rules, the optimal convergence rate is

$$\epsilon_g(n)-\epsilon_{\min}=\frac{A}{\sqrt{n}},\quad n\to\infty. \tag{104}$$

To achieve this rate, λ has to increase as in Eq. (103).

(3) Uniform output noise. In the case of a threshold-smooth system with a rule that is realizable except for a corruption of the labels with probability that is independent of the input, optimal convergence to the underlying teacher rule is achieved for

$$\lambda(n)=\lambda_0 n^2, \tag{105}$$

which yields

$$\epsilon_g(n)-p=\frac{A}{n},\quad n\to\infty. \tag{106}$$

The above results hold for prefactors λ_0 which are smaller than some system-dependent cutoff value.

(4) Realizable rules. In the case of a threshold-smooth system learning a realizable rule, OLGA converges even for finite or zero λ to the teacher weights with an asymptotic convergence rate which is

$$\epsilon_g(n)=\frac{A}{n},\quad n\to\infty. \tag{107}$$

The reason for the remarkable insensitivity to the parameter λ is the fact that according to the update rule, Eq. (45), the magnitude of the change in the weights is proportional to the current value of g. As the system approaches the optimal state, errors occur only when the inputs are near the decision boundary, implying that g vanishes as the zero error limit is approached. Thus our algorithm incorporates a natural tuning of the step size in realizable rules.

In addition to the analysis of the time-dependent λ we have also characterized some of the properties of the equilibrium distributions of weights for fixed large λ. These properties have been derived by expansions near a local minimum of the generalization error. As is well known from general stochastic approximation theory [15–19,30,31], such an analysis is valid for describing the statistics of small fluctuations near the local minima. In addition to these fluctuations there are rare transitions between local minima and these processes which ultimately determine the global shape of the true equilibrium distribution. Our present analysis does not address the issue of the global convergence of the algorithm (except for the special case of realizable rules where the global convergence to the global minimum is guaranteed under fairly general conditions). For the same reason our analysis of the time-dependent λ does not tell to which local minimum of the generalization error the system will converge.

To guarantee convergence to the global minimum of ϵ_g requires in general the addition of noise, namely, the application of OLGA at finite temperature similar to simulated annealing in optimization problems [37–39]. We have shown that in the limit of large λ, the finite-temperature al-

gorithm converges to a Gibbs distribution with the generalization error playing the role of the energy function.

In conclusion, OLGA is an on-line algorithm that has general convergence properties in nonsmooth problems that are similar to those of the stochastic gradient descent algorithm for smooth problems. In both cases, for generic unrealizable rules, if the learning rate is reduced in an appropriate schedule the system converges to a local minimum of ϵ_g. Also in both cases, the addition of an appropriate stochastic component results in a convergence to the global minimum of the generalization error. Since in the case of smooth problems (and large λ) OLGA reduces to the stochastic gradient descent algorithm [see Eq. (35)] OLGA can be considered as a generalization of the standard stochastic gradient descent algorithm that is applicable to a larger class of problems. Finally, comparing our results for OLGA with those of batch learning of threshold functions we note that the asymptotic convergence rate of the on-line algorithm has the same power-law behavior as predicted for batch learning by Vapnik-Chervonenkis (VC) theory [40–43]. It is interesting to note that in the case of the output noise OLGA converges faster than that of batch learning by Gibbs learning, which yields a reduction in error only as the inverse square root of the number of examples, as compared to the inverse power law of Eq. (106) (see also Refs. [44–47]). Although OLGA can yield the optimal power law of reduction in ϵ_g, other algorithms may yield in specific cases faster reduction of errors which will be manifested by a smaller prefactor of the leading power law.

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. This research is partially supported by a grant from the United States–Israel Binational Science Foundation (BSF), Jerusalem, Israel.

APPENDIX A: THE LIMIT OF LARGE λ AT NONZERO T

In this appendix we derive the form of $f(\mathbf{w},\mathbf{s})$ [Eq. (4)] in the large λ limit at nonzero T. We write Eq. (4) as

$$\frac{\beta}{2} f(\mathbf{w},\mathbf{s}) = -\ln \int d\mathbf{x} \exp\left(-\frac{\beta}{4}\|\mathbf{x}\|^2 - \frac{\beta}{2}\epsilon(\mathbf{w}+\mathbf{x}/\sqrt{\lambda},\mathbf{s})\right), \tag{A1}$$

where $\mathbf{x}=\Delta\mathbf{w}\sqrt{\lambda}$, and expand in powers of $1/\sqrt{\lambda}$. For smooth networks, we perform a Taylor expansion of $\epsilon(\mathbf{w},\mathbf{s})$, thereby obtaining after a Gaussian integration over $\mathbf{x}$ and expansion of the logarithmic function,

$$\beta f(\mathbf{w},\mathbf{s}) = N\ln\left(\frac{\beta\lambda}{4\pi}\right) + \beta\epsilon(\mathbf{w},\mathbf{s}) + \frac{1}{\beta\lambda}\left(\beta\nabla^2\epsilon(\mathbf{w},\mathbf{s}) - \frac{\beta^2}{2}\|\nabla\epsilon(\mathbf{w},\mathbf{s})\|^2\right). \tag{A2}$$

In the discrete output case of Eq. (10), we write $\exp(-\beta\epsilon/2) = 1-\epsilon\tau$ where

$$\tau = 1-\exp(-\beta/2). \tag{A3}$$

We then write $\epsilon = \Theta(-\sigma_0 g(\mathbf{w}+\mathbf{x}/\sqrt{\lambda},\mathbf{s}))$ and expand g in powers of $1/\sqrt{\lambda}$. Integrating over $\mathbf{x}$ the result is

$$\beta f(\mathbf{w},\mathbf{s}) = N\ln\left(\frac{\beta\lambda}{4\pi}\right) + \beta\epsilon(\mathbf{w},\mathbf{s}) - 2\epsilon(\mathbf{w},\mathbf{s})\ln[1+\tilde{\tau}H(\sqrt{\beta\lambda/2}\tilde{g}(\mathbf{w},\mathbf{s}))] - 2[1-\epsilon(\mathbf{w},\mathbf{s})]\ln[1-\tau H(\sqrt{\beta\lambda/2}\tilde{g}(\mathbf{w},\mathbf{s}))], \tag{A4}$$

where

$$H(x) = \int_x^\infty \frac{dt}{\sqrt{2\pi}} e^{-(1/2)t^2}, \tag{A5}$$

$$\tilde{\tau} = \exp(\beta/2) - 1, \tag{A6}$$

and

$$\tilde{g}(\mathbf{w},\mathbf{s}) = \frac{|g(\mathbf{w},\mathbf{s})|}{\|\nabla g(\mathbf{w},\mathbf{s})\|}. \tag{A7}$$

Thus in both cases the quenched averaged free energy, defined as

$$f_{\mathrm{av}}(\mathbf{w}) = \langle\langle f(\mathbf{w},\mathbf{s})\rangle\rangle, \tag{A8}$$

has the form

$$f_{\mathrm{av}}(\mathbf{w}) = \frac{N}{\beta}\ln\left(\frac{\beta\lambda}{4\pi}\right) + \epsilon_g(\mathbf{w}) + \delta, \tag{A9}$$

where δ vanishes in the large λ limit. In the smooth network case,

$$\delta \approx -\frac{1}{\beta^2\lambda}\left(\frac{\beta^2}{2}\langle\langle\|\nabla\epsilon(\mathbf{w},\mathbf{s})\|^2\rangle\rangle - \beta\nabla^2\epsilon_g(\mathbf{w})\right). \tag{A10}$$

In the case of discrete outputs,

$$\delta \approx -\frac{1}{\beta\sqrt{\beta\lambda/2}}\langle\langle\delta(g)\|\nabla g\|\rangle\rangle B(\beta), \tag{A11}$$

where

$$B(\beta) = \int_0^\infty dx\{\ln[1-\tau H(x)] + \ln[1+\tilde{\tau}H(x)]\}. \tag{A12}$$

Note that in the discrete case the contribution to δ comes from inputs that lie in the decision boundary of g.

Substituting Eq. (A9) in the average of Eqs. (7) and (8), we obtain

$$\langle\langle\ \langle \Delta \mathbf{w} \rangle\ \rangle\rangle = -\frac{1}{\lambda}\nabla \epsilon_g(\mathbf{w}) + O(\lambda^{-1-\nu}), \tag{A13}$$

$$\langle\langle\ \langle \Delta w^i \Delta w^j \rangle\ \rangle\rangle = \frac{2}{\beta\lambda}\delta_{ij} + O(\lambda^{-1-\nu}), \tag{A14}$$

where $\nu=1$ in the smooth case and $\nu=\frac{1}{2}$ in the threshold-smooth case.

Note: In principle f_{av} provides information about the average connected correlations of $\Delta\mathbf{w}$ and not the full correlations, see Eq. (8). However, in the large λ limit, the disconnected part is of the order of λ^{-2}, since the typical value of $\langle\Delta\mathbf{w}\rangle$ is of the order of λ^{-1}. Therefore the disconnected contribution can be neglected.

APPENDIX B: THE LARGE λ LIMIT FOR NETWORKS WITH OUTPUT NOISE

In this appendix we outline the derivation of the large λ limit of the one-step dynamics of a threshold-smooth system learning a rule corrupted by output noise. Our starting point is Eq. (51), which in terms of the scaled weights, Eq. (64), reads

$$\langle\langle\ \langle \Delta \mathbf{u} \rangle\ \rangle\rangle = \lim_{\lambda\to\infty} \sqrt{\lambda}\langle\langle\ \langle -\tilde{g}(\mathbf{w},\mathbf{s})\nabla\tilde{g}(\mathbf{w},\mathbf{s})\Theta(-\sigma_0\tilde{g}(\mathbf{w},\mathbf{s}))\Theta(\sigma_0\tilde{g}(\mathbf{w},\mathbf{s})+\sqrt{2/\lambda})\rangle\ \rangle\rangle, \tag{B1}$$

where here $\langle\langle\ \rangle\rangle$ is the average over input distribution and $\langle\ \rangle$ is the average over the output noise distribution. Performing the average over the noise distribution, we obtain

$$\langle\langle \Delta \mathbf{u} \rangle\rangle = \lim_{\lambda\to\infty} \sqrt{\lambda}(1-p)\langle\langle -\tilde{g}\nabla\tilde{g}\Theta(-\tilde{g}_0\tilde{g})\Theta(\mathrm{sgn}(\tilde{g}_0)\tilde{g}+\sqrt{2/\lambda})\rangle\rangle + \sqrt{\lambda}p\langle\langle -\tilde{g}\nabla\tilde{g}\Theta(\tilde{g}_0\tilde{g})\Theta(-\mathrm{sgn}(\tilde{g}_0)\tilde{g}+\sqrt{2/\lambda})\rangle\rangle, \tag{B2}$$

where $\tilde{g}=\tilde{g}(\mathbf{w},\mathbf{s})$ and $\tilde{g}_0=\tilde{g}(\mathbf{w}_0,\mathbf{s})$. Expanding $\tilde{g}(\mathbf{w},\mathbf{s})$ about $\mathbf{w}=\mathbf{w}_0$, we obtain $\nabla\tilde{g}\approx\nabla\tilde{g}_0$, and

$$\tilde{g}_0 \approx \tilde{g} - \frac{1}{\sqrt{\lambda}}\nabla\tilde{g}_0\cdot\mathbf{u}. \tag{B3}$$

Defining

$$z = \sqrt{\lambda}\tilde{g} \tag{B4}$$

and

$$v = \nabla\tilde{g}_0\cdot\mathbf{u}, \tag{B5}$$

we can write

$$\Theta(-\tilde{g}_0\tilde{g})\Theta(\mathrm{sgn}(\tilde{g}_0)\tilde{g}+\sqrt{2/\lambda}) = \Theta(v)\Theta(v-z)\Theta(z)\Theta(-z+\sqrt{2}) + \Theta(-v)\Theta(-v+z)\Theta(-z)\Theta(z+\sqrt{2}), \tag{B6}$$

$$\Theta(\tilde{g}_0\tilde{g})\Theta(-\mathrm{sgn}(\tilde{g}_0)\tilde{g}+\sqrt{2/\lambda}) = \Theta(-v+z)\Theta(z)\Theta(-z+\sqrt{2}) + \Theta(v-z)\Theta(-z)\Theta(z+\sqrt{2}). \tag{B7}$$

Finally, integrating over the variable z we obtain

$$\langle\langle \Delta \mathbf{u} \rangle\rangle = -\frac{1}{\sqrt{\lambda}}\mathbf{F}(\mathbf{u}), \tag{B8}$$

where

$$\mathbf{F}(\mathbf{u}) = (1-p)\left\langle\!\!\left\langle \delta(\tilde{g}_0)\nabla\tilde{g}_0 \sum_{\sigma=\pm 1} \sigma\left(\Theta(\sigma v-\sqrt{2}) + \Theta(\sigma v)\Theta(\sqrt{2}-\sigma v)\frac{v^2}{2}\right)\right\rangle\!\!\right\rangle$$
$$+ p\left\langle\!\!\left\langle \delta(\tilde{g}_0)\nabla\tilde{g}_0 \sum_{\sigma=\pm 1} \sigma\Theta(\sigma v+\sqrt{2})\left(-1+\Theta(-\sigma v)\frac{v^2}{2}\right)\right\rangle\!\!\right\rangle. \tag{B9}$$

Note that the average over $\mathbf{s}$ includes averaging over the randomness of v which depends on $\mathbf{s}$ through $\tilde{g}$, see Eq. (B5). Also note that the $\mathbf{u}$ dependence of $\mathbf{F}$ comes through its dependence on v.

In a similar fashion we obtain for the second moment of the single-step dynamics

$$\langle\langle \Delta u_i \Delta u_j \rangle\rangle = \frac{2}{\sqrt{\lambda}} T_{ij}(\mathbf{u}), \tag{B10}$$

where

$$T_{ij}(\mathbf{u}) = (1-p)\left\langle\!\!\left\langle \delta(\tilde{g}_0)\partial_i\tilde{g}_0\partial_j\tilde{g}_0 \sum_{\sigma=\pm 1}\left(\frac{\sqrt{2}}{3}\Theta(\sigma v-\sqrt{2})+\frac{\sigma v^3}{6}\Theta(\sigma v)\Theta(\sqrt{2}-\sigma v)\right)\right\rangle\!\!\right\rangle$$
$$+p\left\langle\!\!\left\langle \delta(\tilde{g}_0)\partial_i\tilde{g}_0\partial_j\tilde{g}_0 \sum_{\sigma=\pm 1}\Theta(\sigma v+\sqrt{2})\left(\frac{\sqrt{2}}{3}+\frac{\sigma v^3}{6}\Theta(-\sigma v)\right)\right\rangle\!\!\right\rangle. \tag{B11}$$

Higher moments of $\Delta\mathbf{u}$ have a similar form. The only λ dependence appears as a prefactor of $1/\sqrt{\lambda}$.

[1] H. Sompolinsky, N. Tishby, and H. S. Seung, Phys. Rev. Lett. **65**, 1683 (1990).
[2] H. S. Seung, H. Sompolinsky, and N. Tishby, Phys. Rev. A **45**, 6056 (1992).
[3] T. L. H. Watkin, A. Rau, and M. Biehl, Rev. Mod. Phys. **65**, 499 (1993).
[4] D. Haussler, M. Kearns, H. S. Seung, and N. Tishby, *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory* (ACM Press, 1994), pp. 76–87.
[5] V. N. Vapnik, *The Nature of Statistical Learning Theory* (Springer-Verlag, Berlin, 1995).
[6] M. Opper and W. Kinzel, in *Physics of Neural Networks III*, edited by J. L. van Hemmen, E. Dommany, and K. Schulten (Springer-Verlag, Berlin, 1996).
[7] H. Sompolinsky, in *Proceedings of the Third NEC Symposium on Computational Learning and Cognition* (SIAM, Princeton, NJ, 1992), p. 217.
[8] S. I. Amari, N. Fujita, and S. Shinomoto, Neural Comput. **4**, 605 (1992).
[9] H. Sompolinsky and N. Barkai, in *Proceedings of the International Joint Conference on Neural Networks* (IEEE, Nagoya, 1993), p. 219.
[10] H. Sompolinsky, N. Barkai, and H. S. Seung, in *Neural Networks: The Statistical Mechanics Perspective*, edited by J. Oh, C. Kwon, and S. Cho (World Scientific, Singapore, 1995).
[11] M. Opper and D. Haussler, Phys. Rev. Lett. **66**, 2677 (1991).
[12] M. Opper and D. Haussler, in *Proceedings of the Fourth Annual Workshop of Computation Learning Theory (COLT91)* (Morgan Kaufmann Publishers, San Mateo, CA, 1991), p. 75.
[13] S. I. Amari, IEEE Trans. Electron. Comput. **16**, 299 (1967).
[14] H. White, J. Am. Stat. Assoc. **84**, 1003 (1989).
[15] T. Heskes and B. Kappen, Phys. Rev. A **44**, 2718 (1991).
[16] G. Radons, H. Schuster, and D. Werner, *International Neural Network Conference '90 Paris* (Kluwer Academic, Dordecht, 1990), p. 993.
[17] L. K. Hansen, R. Pathria, and P. Salamon, J. Phys. A **26**, 63 (1993).
[18] G. Radons, J. Phys. A **26**, 3455 (1993).
[19] T. Heskes, J. Phys. A **27**, 5145 (1994).
[20] M. Biehl and H. Schwarze, J. Phys. A **28**, 643 (1995).
[21] J. K. Anlauf and M. Biehl, Europhys. Lett. **10**, 687 (1989).
[22] M. Biehl and P. Riegler, Europhys. Lett. **28**, 525 (1994).
[23] P. Riegler and M. Biehl, J. Phys. A **28**, 507 (1995).
[24] D. Saad and S. Solla, Phys. Rev. Lett. **74**, 4337 (1995).
[25] D. Saad and S. Solla, Phys. Rev. E **52**, 4225 (1995).
[26] T. Heskes, E. T. P. Slijpen, and B. Kappen, Phys. Rev. A **46**, 5221 (1992).
[27] N. Barkai, H. S. Seung, and H. Sompolinsky, Phys. Rev. Lett. **75**, 1415 (1995).
[28] 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.
[29] N. Barkai, Ph.D. thesis, Hebrew University of Jerusalem, 1995.
[30] H. J. Kushner and D. S. Clark, *Stochastic Approximation Methods for Constrained and Unconstrained Systems* (Springer-Verlag, Berlin, 1978).
[31] N. G. van Kampen, *Stochastic Processes in Physics and Chemistry* (North-Holland, Amsterdam, 1981).
[32] J. W. Kim and H. Sompolinsky, Phys. Rev. Lett. **76**, 3021 (1996).
[33] J. W. Kim and H. Sompolinsky, Phys. Rev. Lett. **78**, 4306 (1997).
[34] M. L. Minsky and S. A. Papert, *Perceptrons* (MIT Press, Cambridge, MA, 1969).
[35] N. J. Nilsson, *Learning Machines* (McGraw-Hill, New York, 1965); G. J. Mitchison and R. M. Durbin, Biol. Cybern. **60**, 345 (1989).
[36] J. Kivinen and M. K. Warmuth, J. Inform. Comput. **132**, 1 (1997).
[37] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, Science **220**, 671 (1983).
[38] S. Geman and D. Geman, IEEE Trans. Pattern Anal. Mach. Intell. **6**, 721 (1984).
[39] H. J. Kushner, SIAM (Soc. Ind. Appl. Math.) J. Appl. Math. **47**, 169 (1987).
[40] V. N. Vapnik and A. Y. Chervonenkis, Theor. Probab. Appl. **16**, 264 (1971).
[41] V. N. Vapnik, *Estimation of Dependences Based on Empirical Data* (Springer-Verlag, Berlin, 1982).
[42] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth,

J. Assoc. Comput. Mach. **36**, 929 (1989).
[43] A. Ehrenfeucht, D. Haussler, M. A. Kearns, and L. Valiant, Inform. Comput. **82**, 247 (1989).
[44] C. van den Broeck and P. Reimann, Phys. Rev. Lett. **76**, 2188 (1996).
[45] M. Opper, Phys. Rev. Lett. **77**, 4671 (1996).
[46] M. Biehl, P. Riegler, and M. Stechert, Phys. Rev. E **52**, R4625 (1995).
[47] M. Copelli, O. Kinouchi, and N. Caticha, Phys. Rev. E **53**, 6341 (1996).