Learning rate and attractor size of the single-layer perceptron

Martin S. Singleton*
Department of Mathematics, University of Illinois at Urbana-Champaign, Urbana, Illinois 61801, USA

Alfred W. Hübler†
Department of Physics, University of Illinois at Urbana-Champaign, Urbana, Illinois 61801, USA

(Received 12 August 2006; revised manuscript received 5 January 2007; published 26 February 2007)

We study the simplest possible order one single-layer perceptron with two inputs, using the delta rule with online learning, in order to derive closed form expressions for the mean convergence rates. We investigate the rate of convergence in weight space of the weight vectors corresponding to each of the 14 out of 16 linearly separable rules. These vectors follow zigzagging lines through the piecewise constant vector field to their respective attractors. Based on our studies, we conclude that a single-layer perceptron with N inputs will converge in an average number of steps given by an Nth order polynomial in $\frac{t}{l}$, where t is the threshold, and l is the size of the initial weight distribution. Exact values for these averages are provided for the five linearly separable classes with $N=2$. We also demonstrate that the learning rate is determined by the attractor size, and that the attractors of a single-layer perceptron with N inputs partition $\mathbb{R}^N \oplus \mathbb{R}^N$.

DOI: 10.1103/PhysRevE.75.026704 PACS number(s): 07.05.Mh, 05.45.−a, 84.35.+i, 87.18.Sn

I. INTRODUCTION

Since its advent in 1960, the study of the perceptron and of neural networks has experienced vast growth. Some fundamental characteristics about perceptrons were studied initially, while more recently several sophisticated and subtle questions have been addressed by the physics community. One significant current of research is in online learning, where amongst other aspects that of ensemble learning (the combination of many different learning rules) of simple perceptrons has been examined [1]. More recently there has been an interest in studying the effects of noise on learning in recurrent perceptron networks [2], and aspects of convergence for perceptrons with stochastic, binary synapses [3]. One of the tools used to analyze perceptron generalization and capacity is the replica method [4], and this technique has been used to determine the optimal capacity of ternary perceptrons [5], the generalization and capacity of large two-layered perceptrons [6], as well as to study learning capability under mutual information maximization [7].

Many other techniques for studying neural networks have appeared in the recent physics literature. For instance, one important area of current physics research treats the issue of generating and learning time series, including chaotic time series, by perceptrons [8–11]. The study of time series generation by the perceptron is notable in addition as they have been used to produce limit cycles [11]. Information theory approaches have also been fruitful in yielding storage capacity values for neural networks with binary weights in good agreement with the replica method [12], as well as an information gain principle which can give insight on how to choose training sets and transfer functions for student-teacher learning paradigm perceptrons [13]. Moreover, it is remarkable that perceptron learning has also been applied to biophysics problems, in particular in the context of improving the pairwise contact energy function in the study of the protein folding problem [14].

There are numerous convergence theorems which demonstrate that a perceptron learning a linearly separable rule will converge in a finite number of steps [15]. Recently, analogous upper bounds for the number of steps have been proven for continous perceptrons using online gradient learning [16] and Boolean multilinear perceptrons [17]. In addition, systematic studies have been made of the rate of convergence for perceptrons learning nonseparable rules [18], and order of magnitude comparisons were given for learning rates for batch, online, and cyclic learning [19].

There have been, however, no studies of visualizing the convergent weight dynamics of perceptrons, and very few in studying quantitatively the number of steps required to converge within weight space. In this paper we address these issues by giving concrete results from direct calculations, which quantify precisely the mean number of steps the weight vector travels in weight space according to which (linearly separable) rule it is learning. Our paper is organized as follows. In the next section, we discuss in detail the exact algorithm we used to obtain our results, and explain how the perceptron learning procedure itself necessarily fills weight space completely with its attractors. In the section on convergence rates, we outline our calculation for the average convergence rates, and state these for $N=2$. Finally we discuss some implications of these results, and in the conclusion offer some possible applications and directions for future research.

II. PERCEPTRON ALGORITHM

We use a cyclic online binary perceptron algorithm [19], which we outline as follows. A binary perceptron implements

*Electronic address: martin@math.uiuc.edu

†Permanent address: Center for Complex Systems Research, Department of Physics, 1110 W Green Street, University of Illinois at Urbana-Champaign, IL 61801, USA. Electronic address: a-hubler@uiuc.edu

a Boolean function which maps $\mathbb{Z}_2^N$ to $\mathbb{Z}_2$. The input vectors are represented by the real numbers $x^j = x_1 2^{N-1} + x_2 2^{N-2} + \cdots + x_N$, where $x_i \in \{0,1\}$, and $0 \leqslant j \leqslant 2^N - 1$. There are 2^N possible input patterns (vectors) since each component can either be 0 or 1. The weight vectors are written as $w_n^k \in \mathbb{R}^N, 0 \leqslant n < \infty$, where k corresponds to a rule, to be defined below. Here n is to be thought of as the discrete time, and there are 2^N weight components since each input (a component of the input vector) is weighted by a corresponding component of the weight vector. The desired output for each of the 2^N input vectors x^j is denoted by $d_j \in \mathbb{Z}_2$, $0 \leqslant j \leqslant 2^N - 1$. The actual output of the neuron y is given by

$$y(w_n^k, x^j) = \Theta(w_n^k x^j + (-1)^{\lceil k/(2^{2^N-1})\rceil} t), \quad 0 \leqslant k \leqslant 2^{2^N} - 1. \tag{1}$$

Here $(-1)^{\lceil k/(2^{2^N-1})\rceil} t$ is the threshold ($\lceil z \rceil$ denotes the smallest integer greater than or equal to z) and $\Theta(x)$ is the Heaviside function, so that the perceptron learning algorithm can be thought of as a nonlinear map $\mathcal{T}: \mathbb{R}^N \to \mathbb{R}^N$. The meaning of Eq. (1) is that the existing weight configuration w_n^k specifies the normal to a hyperplane $\mathcal{H} = \{x \in \mathbb{R}^N | w_n^k x + (-1)^{\lceil k/(2^{2^N-1})\rceil} t = 0\}$. Whether $y(w_n^k, x^j)$ is one or zero thus determines whether the perceptron currently "perceives" that the input vector x^j should be on one side of the hyperplane or the other. Thus learning in this context means the perceptron embodies a plane which properly separates the input vectors into two classes, which fall on either side of the plane. Indeed, there is a set of such hyperplanes, each one corresponding to a "rule" which the perceptron can learn. A rule specifies an output for each of the 2^N input vectors. $d^k = \{d_j^k\}_{0 \leqslant j \leqslant 2^N - 1}, 0 \leqslant k \leqslant 2^{2^N} - 1$ can be thought of as a 2^N component vector which describes a rule (or function) k which the perceptron should learn, since its components are the desired outputs corresponding to each of the inputs. Since the perceptron's output is either 0 or 1, there are thus 2^{2^N} possible rules to consider for the perceptron to learn.

At time $n=0$ one of these 2^{2^N} rules is selected and the weight vectors are initialized to random values with components uniformly distributed in the N-dimensional hypercube centered at the origin, with faces orthogonal to the coordinate axes, and sides of length $2l$, in the weight space $\mathbb{R}^N$. For each time step n, each of the 2^N input vectors is presented in the cyclic online order. Weight vectors are changed according to the prescription:

$$w_{n+1}^k = w_n^k + a[d_j^k - y(w_n^k, x^j)]x^j, \quad 0 \leqslant j \leqslant 2^N - 1, \tag{2}$$

where a is the adaptation (learning) rate, and $j = n \bmod 2^N$.

Of the 2^{2^N} possible functions to be learned, only a fraction will be learnable. From Eq. (2), the rule k has been learned if the weights stop changing, which is equivalent to the condition that there exists $n_0 \in \mathbb{Z}^+$ such that $w_{n+1} = w_n$, for all $n \geqslant n_0$. In this case $y(w_n^k, x^j) = d_j^k$, $0 \leqslant j \leqslant 2^N - 1$, and thus the learning algorithm terminates. If the rule k is not learnable, then the algorithm never terminates, i.e., $\lim_{n\to\infty} w_n^k$ does not exist. We discuss this matter further below in connection with the XOR rule. There is also an equivalent theoretical notion of learnability, which we state for completeness. Let the input patterns be separated into two sets, according to whether the output should be 0 (C^0) or the output should be 1 (C^1). We say a rule $k, 0 \leqslant k \leqslant 2^{2^N} - 1$ is learnable, or "linearly separable" if there exists a number $\delta > 0$ and a weight (suppressing subscripts and superscripts momentarily) $w^* \in \mathbb{R}^N$ such that

$$w^* x < -\delta, \quad \text{if } x \in C^0 \tag{3}$$

and

$$w^* x > \delta, \quad \text{if } x \in C^1. \tag{4}$$

Note that the definition of linear separability is equivalent to the existence of a hyperplane $\mathcal{H}^k \in \mathbb{R}^{N-1}$ which separates the two sets of input vectors according to whether the desired output is 0 or 1. Indeed, to see this one takes w^* as the normal to the hyperplane, then the input vectors as points in $\mathbb{R}^N$ fall on either side of $\mathcal{H}^k$ by the equation for $\mathcal{H}^k: w^* x = 0$.

We now specialize to the case of $N=2$, in order to derive exact formulas for the average convergence rates. A rule k (Boolean function) is identified by the vector of its four outputs $d^k = (d_0^k, d_1^k, d_2^k, d_3^k)$, corresponding to input vectors (0,0), (0,1), (1,0), and (1,1), respectively. For $N=2$, 14 of 16 possible functions are linearly separable. By the symmetry of the patterns, the number of classes, based on the average number of steps to converge, is five. There are two attractors in class I [$d^0 = (0,0,0,0)$ and $d^{15} = (1,1,1,1)$], two attractors in class II [$d^1 = (0,0,0,1)$ and $d^{14} = (1,1,1,0)$], four attractors in class III [$d^2 = (0,0,1,0)$, $d^4 = (0,1,0,0)$, $d^{11} = (1,0,1,1)$, and $d^{13} = (1,1,0,1)$], four attractors in class IV [$d^3 = (0,0,1,1)$, $d^5 = (0,1,0,1)$, $d^{10} = (1,0,1,0)$, and $d^{12} = (1,1,0,0)$], and two attractors in class V [$d^7 = (0,1,1,1)$ and $d^8 = (1,0,0,0)$]. We can see why the rules have the above class structure by the following symmetry considerations. If $d^2 = d^3$, we say the rule is symmetric, otherwise the rule is antisymmetric. If the rule is antisymmetric, then interchanging d^2 and d^3 gives a different rule (which we say is antisymmetric to the original rule) with equivalent dynamics, which yields a factor of 2. We say that a rule $\tilde{k}$ has equivalent weight dynamics to a rule k if given initial conditions that are reflected across some line or point of symmetry, the average trajectories of the weights stay reflected in the same way for the two different rules. The components of the average trajectories, $\langle w_{i,n} \rangle$, are defined by

$$\langle w_{i,n} \rangle \equiv \frac{1}{4} \sum_{m=n}^{n+3} w_{i,m}, \quad \text{where } 1 \leqslant i \leqslant N. \tag{5}$$

In this case, it follows that if two different rules have equivalent dynamics, and the initial conditions are isotropic, then the weight vectors under both rules exhibit the same statistical convergence properties. We show in Appendix A (appendix statement A.1) that if two rules k and $\tilde{k}$ are antisymmetric, and if the initial conditions are antisymmetric (initial weight vector reflected across line $w_1 = w_2$) then

$$\begin{pmatrix} w^k_{1,4m} \\ w^k_{2,4m} \end{pmatrix} = \begin{pmatrix} w^{\tilde{k}}_{2,4m} \\ w^{\tilde{k}}_{1,4m} \end{pmatrix}, \quad m = 0,1, \ldots . \tag{6}$$

Thus two antisymmetric rules with antisymmetric initial conditions have trajectories which coincide every fourth step in the algorithm. In fact, according to statement A.3 of Appendix A, the average dynamics for antisymmetric rules k and $\tilde{k}$ are antisymmetric given antisymmetric initial conditions:

$$\begin{pmatrix} \langle w^k_{1,n} \rangle \\ \langle w^k_{2,n} \rangle \end{pmatrix} = \begin{pmatrix} \langle w^{\tilde{k}}_{2,n} \rangle \\ \langle w^{\tilde{k}}_{1,n} \rangle \end{pmatrix}, \quad n = 0,1, \ldots . \tag{7}$$

If two rules k and $\tilde{k}$ are opposite (rule k is the opposite of rule $\tilde{k}$ if $\tilde{d}^k = d^{\tilde{k}}$), then the weight dynamics will also be equivalent under the two rules. In particular, in Appendix A it is shown (appendix statement A.3) that if two rules k and $\tilde{k}$ are opposite and the corresponding initial conditions for the weights are reflected through the origin, then the average trajectories themselves will be reflected through the origin for the two rules, i.e.,

$$\begin{pmatrix} \langle w^k_{1,n} \rangle \\ \langle w^k_{2,n} \rangle \end{pmatrix} = - \begin{pmatrix} \langle w^{\tilde{k}}_{1,n} \rangle \\ \langle w^{\tilde{k}}_{2,n} \rangle \end{pmatrix}, \quad n = 0,1, \ldots . \tag{8}$$

Thus this yields an additional factor of 2 for the number of members of each rule class. Hence each class of symmetric rules, of which there are three, has two members, and each class of antisymmetric rules has four members. This accounts for all 14 learnable of 16 possible rules.

The XOR rule, $d^6 = (0,1,1,0)$ and its opposite (1, 0, 0, 1) are not learnable, which can be seen by trying to separate by one line the points (1, 0) and (0, 1) in one class from (0, 0) and (1, 1) in the other. This also follows from a sufficient condition for unlearnability. Let x^i and x^j be two vectors in one of the classes to be separated, say C_0. It follows that if they are both on the same side of the separating hyperplane, the line L through these points must lie completely on that side of the hyperplane. It follows that the classification is unlearnable if some other line L' connecting two points in C_1 intersects the line L. For the case of the XOR rule, take $x^2 = (0,1)$ and $x^3 = (1,0)$, then the line L through x^2 and x^3 interesects the line L' connecting the two points (0, 0) and (1, 1) from the other class at the point $(1/2, 1/2)$ so that the rule is not learnable. Thus there are three symmetric rule classes, with two elements in each class. This accounts for all 14 learnable of 16 possible rules. Note that if the distance between two randomly situated initial weight vectors were fixed at random values and this distance were observed over time for the perceptron attempting to learn the XOR rule, this value would cycle periodically with various small periods. This indicates that the XOR rule does not have a chaotic attractor.

Calculating the weight vectors for the five representatives of the convergent Boolean functions, the solutions of the resulting linear inequalities are shown in Fig. 1. As expected, these figures, which are the "attractors" for the weight dynamics, correspond exactly to the attractors found by the perceptron convergence algorithm in Fig. 2. In Table I we give the linear inequalitities for each of the five class representatives, corresponding to Fig. 1. In general, these inequalities for any of the 16 rules take the form

$$w_2(-1)^{d^1} \leqslant t(-1)^{d^1}, \quad w_1(-1)^{d^2} \leqslant t(-1)^{d^2},$$

$$w_2(-1)^{d^3} \leqslant (-w_1 + t)(-1)^{d^3}. \tag{9}$$

From solving the linear inequalities for all 14 attractors we see that the attractors partition $\mathbb{R}^2$ twice as required for $N = 2$.

III. CONVERGENCE RATES

Consider the learnable functions (classifications) for the single-layer perceptron with N inputs. We fix the threshold at $\pm t$, where $t > 0$ is small. Then each convergent classification k corresponds to a finite or infinite attractor $A_k \in \mathbb{R}^N$ for the weight dynamics. The attractor A_k is a convex set (a cone) formed by the intersection of two or more hyperplanes. Given an initial weight probability distribution with probability density function $p(x)$, the expected number μ_k of weight changes to converge is given by

$$\mu_k = \int_{\mathbb{R}^N} p(x)\,\eta(A_k,x)dx, \tag{10}$$

where $\eta(A_k, w)$ is essentially proportional to the length of the path from the initial weight vector w to the attractor A_k. Similarly, the variance σ_k of the expected number of weight changes is given by

$$\sigma_k^2 = \int_{\mathbb{R}^n} p(x)\,\eta(A_k,x)^2 dx - \mu_k^2. \tag{11}$$

To find $\eta(A_k, x)$, we recall from Eq. (2) that the average trajectory of the weight vector is determined by the map:

$$w_{n+1} = w_n + ac_R, \tag{12}$$

where

$$c_R = \frac{1}{2^N} \sum_{j=1}^{2^N} [d^k_j - y(\langle w^k \rangle_R, x^j)] x^j, \tag{13}$$

and $\langle w^k \rangle_R$ denotes a range of weight vectors within the region R which produce the same output for y. Thus given that the weight vector's path will be determined by the initial position of the weight vector $x = w_0$ and the rule k, the form $\eta(A_k, x)$ takes is

$$\eta(A_k,x) = \sum_{R \in \mathbb{R}^N} r_R d_R, \tag{14}$$

where d_R is a distance computed from the direction c_R of the trajectory in R and the location of the next region, and r_R is a rate equal to the number of steps per unit distance traveled

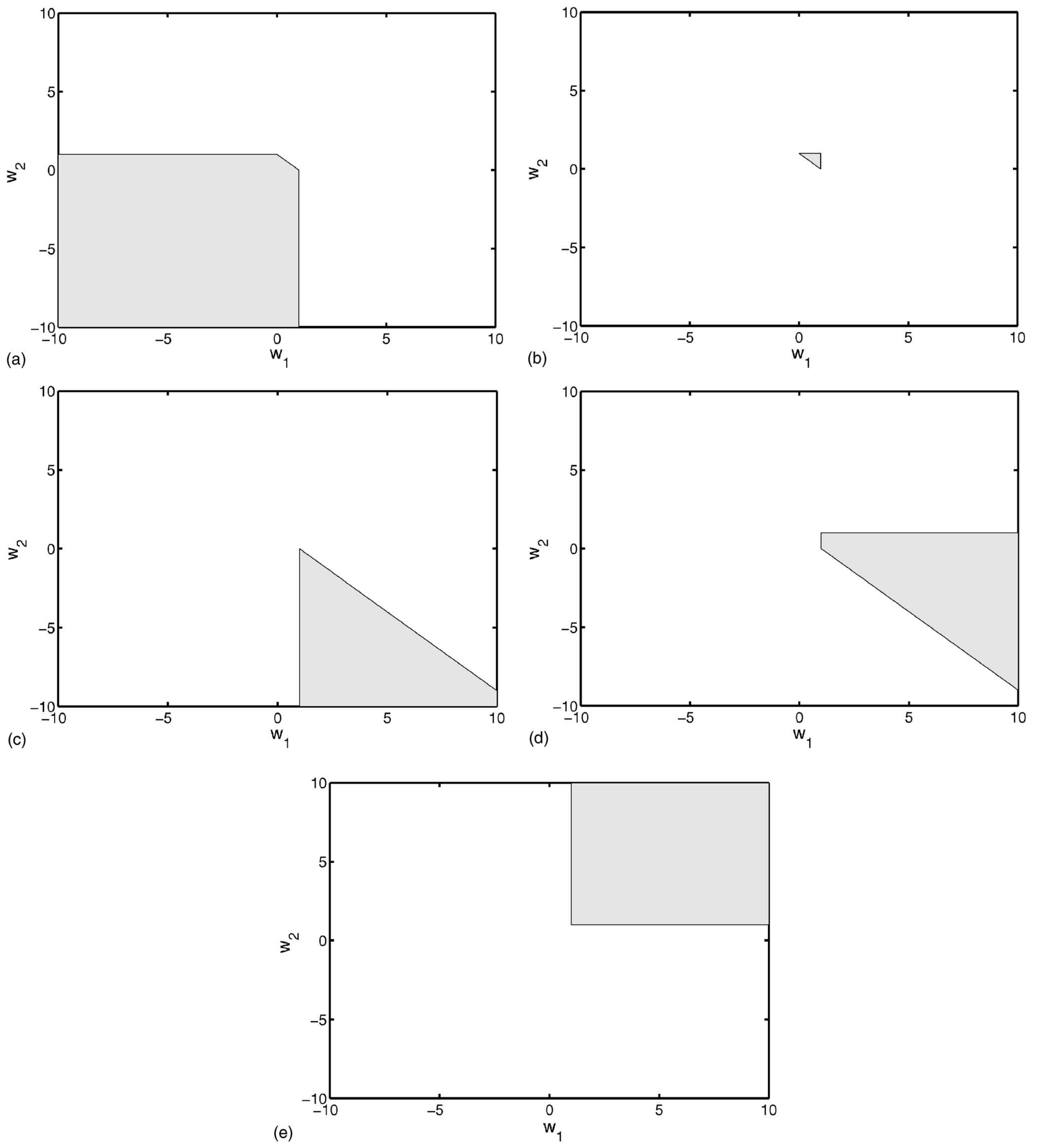

FIG. 1. The solutions of the linear inequalities in Table I, corresponding to the attractors of the five learnable classes for the single-layer perceptron. (a) class I representative (0,0,0,0), (b) class II representative (0,0,0,1), (c) class III representative (0,0,1,0), (d) class IV representative (0,0,1,1), and (e) class V representative (0,1,1,1), Note that these regions coincide exactly with the attractors in weight space shown in Fig. 2.

by the weight vector. The rate r_R can be determined from a and observing which values of j, $1 \leqslant j \leqslant 2^N$, produce weight changes in region R, and it measures zigzagging on a finer level than that of the weights as they traverse the different regions. Since some trajectories travel close along the boundaries of the regions, the rate r_R was sometimes based on a best estimate of the trajectory at this finest level. In Appendix B we give an example calculation of Eq. (20) by using Eqs. (10) and (12).

In our investigation, $N=2$, and we use a uniform probability distribution of initial weight vectors on a square of side length $2l$:

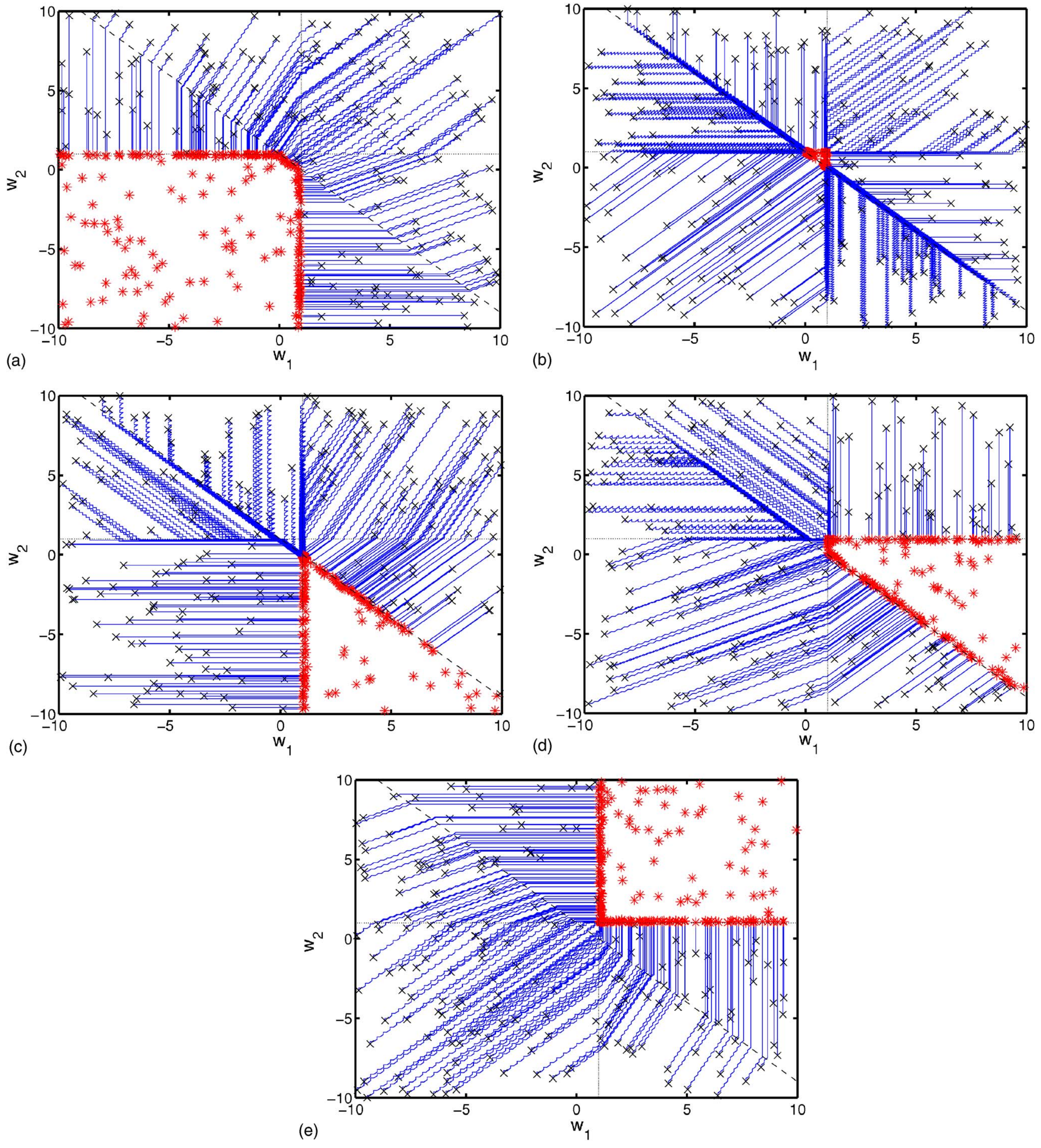

FIG. 2. (Color online) The paths in weight space for 250 trials of the converging weight vectors. ×: beginning point of trajectory. *: end point of trajectory. (a) class I representative (0,0,0,0), $\mu=24.47$, $\sigma=22.19$, (b) class II representative (0,0,0,1), $\mu=76.33$, $\sigma=40.80$, (c) class III representative (0,0,1,0), $\mu=43.59$, $\sigma=34.34$, (d) class IV representative (0,0,1,1), $\mu=41.44$, $\sigma=31.04$, and (e) class V representative (0,1,1,1), $\mu=37.91$, $\sigma=27.04$. By calculating the mean and standard deviation of the number of steps over these paths, we arrived at expressions (16)–(25).

$$p(x)=\frac{1}{4l^2}1_{(|x|\leq l,|y|\leq l)}. \tag{15}$$

Computing μ_k and σ_k^2 for $k=1$ to 5 from Eq. (10) and (11) for the five function classes, we find:

$$\mu_1(a,l,t)=\frac{22l^3-30l^2t+30lt^2-49t^3}{48al^2}, \tag{16}$$

TABLE I. Inequalities corresponding to attractors of Fig. 1, for the representitives of the five function classes. In Fig. 1 the threshold is set to $-t=-1$.

Outputs	Attractor inequalities
(0,0,0,0)	$w_2 \leqslant t,\ w_1 \leqslant t,\ w_2 \leqslant -w_1+t$
(0,0,0,1)	$w_2 \leqslant t,\ w_1 \leqslant t,\ w_2 \geqslant -w_1+t$
(0,0,1,0)	$w_2 \leqslant t,\ w_1 \geqslant t,\ w_2 \leqslant -w_1+t$
(0,0,1,1)	$w_2 \leqslant t,\ w_1 \geqslant t,\ w_2 \geqslant -w_1+t$
(0,1,1,1)	$w_2 \geqslant t,\ w_1 \geqslant t,\ w_2 \geqslant -w_1+t$

$$\sigma_1^2(a,l,t) = \frac{1}{2304a^2l^4}(356l^6 - 504l^5t + 804l^4t^2 + 980l^3t^3 - 5628l^2t^4 + 2940lt^5 - 2401t^6), \tag{17}$$

$$\mu_2(a,l,t) = \frac{30l^3 - 30l^2t + 27lt^2 - 11t^3}{24al^2}, \tag{18}$$

$$\sigma_2^2(a,l,t) = \frac{1}{576a^2l^4}(204l^6 + 72l^5t - 864l^4t^2 + 1536l^3t^3 - 1317l^2t^4 + 594lt^5 - 121t^6), \tag{19}$$

$$\mu_3(a,l,t) = \frac{103l^3 + 93l^2t + 97lt^2 + 7t^3}{192al^2}, \tag{20}$$

$$\sigma_3^2(a,l,t) = \frac{1}{110\,592a^2l^4}(10\,725l^6 - 90\,018l^5t + 207\,435l^4t^2 - 188\,788l^3t^3 + 44\,275l^2t^4 - 4074lt^5 - 147t^6), \tag{21}$$

$$\mu_4(a,l,t) = \frac{113l^3 + 3l^2t + 93lt^2 - 7t^3}{192al^2}, \tag{22}$$

$$\sigma_4^2(a,l,t) = \frac{1}{36\,864a^2l^4}(8135l^6 + 2874l^5t + 5901l^4t^2 - 2336l^3t^3 - 4119l^2t^4 + 1302lt^5 - 49t^6), \tag{23}$$

$$\mu_5(a,l,t) = \frac{44l^3 + 84l^2t + 42lt^2 - t^3}{96al^2}, \tag{24}$$

$$\sigma_5^2(a,l,t) = \frac{1}{9216a^2l^4}(1424l^6 + 2976l^5t + 1056l^4t^2 - 1112l^3t^3 - 720l^2l^4 + 84lt^5 - t^6). \tag{25}$$

Following the logic of the above calculation, a generalization shows that the mean number of steps to converge for a linearly separable rule k for the general case of an arbitrary number of inputs N is a polynomial in $\frac{t}{l}$:

$$\mu_k\left(\frac{t}{l};\frac{l}{a}\right) = \frac{l}{a}\sum_{j=0}^{N+1} a_j\left(\frac{t}{l}\right)^j, \tag{26}$$

treating $\frac{l}{a}$ as a parameter, where the $N+1$ coefficients can in principle be calculated. Thus by using an appropriate form of optimization to calculate the coefficients a_j in Eq. (26), our method can be extended to yield the mean polynomial time convergence expressions for a perceptron with an arbitrary number N of inputs.

Note that the formulas we give here are valid within the range $a<t<l$, which is a reasonable and normal ordering for these parameters. In particular our simulations were performed taking the values $a=0.15$, $t=1$, and $l=10$. If any of these parameters were to approach equality with any of the others, it is not expected that the above formulas would continue to be valid. Thus these results come with the caveat that they should only be used for "typical" ranges of perceptron neural networks.

IV. RESULTS

Figure 3 shows the histograms of the number of steps to converge for the five pattern classes, using simulations of 10 000 trials. The corresponding means and variances as calculated from Eqs. (16)–(25) are given in Table II. (In the simulation and Table II we give the standard deviation, for its greater qualitative meaningfulness, instead of the variance.) We find that the simulation results agree very well with the values given by the calculated expressions for the means and variances. In particular, four of the five means calculated agree within a couple of standard errors of the mean (SEM) with the simulation results. Only one rule, (0, 0, 1, 0) deviated significantly between predicted (39.33) and simulated (44.39) mean number of steps. This is most likely due to some error in the calculation of the number of steps along boundaries of the regions [cf. Fig. 2].

Figure 2 shows the paths of the weight vectors for 250 trials. In some of these it is clear that a path has a maximum of three directions, each of which is a straight line (though on a finer scale this is a zigzag). This is a consequence of the form taken by the piecewise constant vector field c_R [Eq. (13)]. Only three of the input weight vectors are nonzero, thus there will be a maximum of three directions. The trajectories are piecewise linear since multiples of the input vectors are always added to the weight vector [cf. Eq. (12)].

V. DISCUSSION

From our method of calculation, it follows that the smaller the attractor, the slower in general the rate of convergence. This is illustrated by Table I and Fig. 1. On the other hand, one can interpret our results as showing that some inputs to the network result in "resonance" [20,21] in the learning rate. In particular the class of functions corresponding to the outputs (0,0,1,0) and (0,1,0,0) were found to be resonant under this interpretation, if we disregard the "output always off" rule, (0,0,0,0). In a previous study, the order of magnitude of convergence rate is given as $\sqrt{P/\epsilon}\log(1/\epsilon)$, where P is the number of patterns and ϵ is

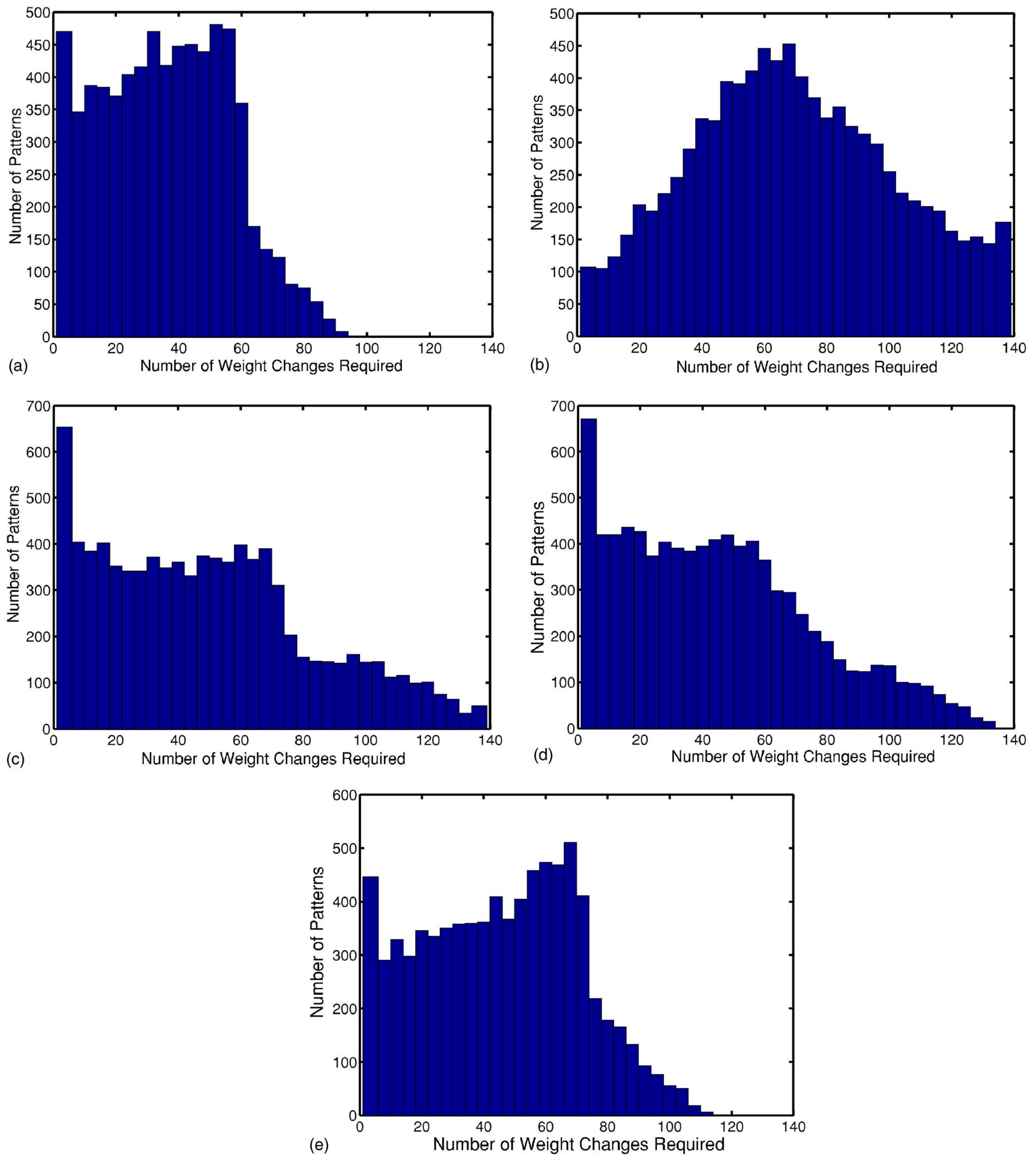

FIG. 3. (Color online) The histograms of the five different classes for 10 000 trials are shown. (a) class I representative (0,0,0,0), $\mu=25.77$, $\sigma=23.98$, (b) class II representative (0,0,0,1), $\mu=77.50$, $\sigma=40.06$, (c) class III representative (0,0,1,0), $\mu=44.39$, $\sigma=35.58$, (d) class IV representative (0,0,1,1), $\mu=40.24$, $\sigma=32.45$, and (e) class V representative (0,1,1,1), $\mu=37.05$, $\sigma=29.22$. Note that four classes have fairly similar shapes, having their distributions weighted towards the left, while the "AND" rule is centrally distributed. Trials which were already in the attractors at $n=0$ were omitted for clarity.

the adaptation [19]. This gives for our values of P and ϵ, the number of learning steps as 21. As shown above, our results are more precise and they distinguish between the different learning rules which the earlier studies do not.

Another improvement of our results over previous studies of perceptron learning performance is our closed form calculation of the variances of the convergence rates. Recent work in the field of computational complexity has shown that hard

TABLE II. Predicted and observed means and standard deviations of the number of steps to converge from Eqs. (16)–(25) and 10 000 trials. To compare note that, e.g., the standard error of the mean for (0,0,0,0) would be $23.98/\sqrt{10\,000}=0.24$.

	Theory		Simulation	
Rule	Mean	Standard deviation	Mean	Standard deviation
(0,0,0,0)	26.74	24.61	25.77	23.98
(0,0,0,1)	75.72	39.67	77.50	40.06
(0,0,1,0)	39.33	12.05	44.39	35.58
(0,0,1,1)	39.66	31.97	40.24	32.45
(0,1,1,1)	36.68	28.89	37.05	29.22

computational problems may better be tackled by combining several different approaches [22], thus detailed knowledge of learning performance of each component algorithm is crucial. It is interesting to note in connection with these findings of polynomial (in the variable $\frac{t}{l}$) time convergence rates on average the results for P complexity of single-layer perceptrons learning linearly separable rules, and NP-complete complexity of two-layer perceptrons [23]. Indeed, our formulas suggest that the problem of a single-layer perceptron converging has polynomial complexity, by direct computation of such polynomials. (Since our expressions give only averages we cannot claim they guarantee polynomial time convergence.) This dynamic between the single-layer perceptron polynomial time convergence versus the two-layer perceptron nondeterministic polynomial time convergence evokes a recent result which showed that NP-complete problems exhibit phase boundaries [24] away from which the problems become easier. Thus (considering a quasicontinuous case) inputs which transition from the "XOR" rule to the "AND" rule may be seen to undergo an analogous phase transition in difficulty. We therefore suggest that a greater understanding of the relation between P and NP-complete computational complexity classifications might be obtained by generalizing our computational approach for two-layer neural networks, though it is not clear to us at present how to proceed in this undertaking.

By the same reasoning which led to the result that inequalities (9) give 14 attractors which partition $\mathbb{R}^2$ twice, it follows that a similar (though much larger) set of inequalities corresponding to the linearly separable Boolean functions of a perceptron with N inputs partitions $\mathbb{R}^N \oplus \mathbb{R}^N$. In particular, the perceptron must solve the system:

$$a_{11}w_1 + a_{12}w_2 + \cdots + a_{1N}w_N \geqslant t,$$

$$a_{21}w_1 + a_{22}w_2 + \cdots + a_{2N}w_N \geqslant t,$$

$$\cdots\ \cdots$$

$$a_{k1}w_1 + a_{k2}w_2 + \cdots + a_{kN}w_N \geqslant t,$$

$$a_{(k+1)1}w_1 + a_{(k+1)2}w_2 + \cdots + a_{(k+1)N}w_N \leqslant t,$$

$$\cdots\ \cdots$$

$$a_{2^N1}w_1 + a_{2^N2}w_2 + \cdots + a_{2^NN}w_N \leqslant t, \tag{27}$$

where $1 \leqslant k \leqslant 2^N$, $-t$ is the numerical value of the threshold (for the higher order rules the threshold will be t), and the inequalities were rearranged so that all $\geqslant$ appear before any $\leqslant$, and $a_{ij} \in \{0,1\}$, $1 \leqslant i \leqslant 2^N$, $1 \leqslant j \leqslant N$. As with a system of 2^N linear equations in N unknowns, there will either be an infinite number of solutions to this system, or no solution. In this connection there is a useful question to ask: is there a deterministic calculation which can always establish, as a function of k and the coefficients a_{ij}, whether or not the system of inequalities (27) (and thus could the perceptron learn the corresponding rule) has a solution? We do not have an answer for this, though we suggest that the answer would be yes. Nevertheless we can see from the above generalization that the set of attractors for the single-layer perceptron with N inputs is isomorphic to $\mathbb{R}^N \oplus \mathbb{R}^N$. This leads to the question of how many separate regions into which $\mathbb{R}^N$ can be divided.

In the general case of a perceptron with N inputs, we have a situation of hyperplanes which partition $\mathbb{R}^N$ into regions in which the weight dynamics follows straight lines [see Eq. (12)]. If $N=2$, then the hyperplanes are lines, and they partition $\mathbb{R}^2$ in the general case into seven regions. For $N=3$, the hyperplanes are planes in $\mathbb{R}^3$, and in this case we found that they divide $\mathbb{R}^3$ into 26 regions, as there were 26 attractors. It is interesting to speculate on how many regions 2^{N-1} hyperplanes in $\mathbb{R}^N$ can divide $\mathbb{R}^N$ into in the most general case, since this number would give an upper bound on the number of rules which a single-layer perceptron with N inputs could learn. A simple counting argument shows that in $\mathbb{R}^2$ there are again seven such regions. In this connection a result by Polya [25] gives as 64 the number of regions which seven planes divide $\mathbb{R}^3$ up into in the most general case, which indicates that perceptrons do not partition space in the most general way.

Note that for the general case of $\mathbb{R}^N$, our sufficient condition for nonlearnability of a classification can be restated in terms of hyperplanes. In particular, let $\{P_1, \ldots, P_r\}$ and $\{Q_1, \ldots, Q_s\}$ be points in the classes C_0 and C_1 and denote their respective convex hulls by S_0 and S_1. Then if $S_0 \cap S_1 \neq \emptyset$, the classification is not learnable.

VI. CONCLUSION

Our main result is the derivation of exact averages and variances of the convergence rates for $N=2$ perceptrons, and the general polynomial form Eq. (26) of the average convergence rates for perceptrons with N input vectors. Possible future work might, beyond extending these results to include multiperceptrons as mentioned above, include algorithms or methods for exact calculations of the coefficients for the polynomials given in Eq. (26). We suggest that similar mean convergence rate results may also be obtainable by generalizing this approach to other types of neural networks frequently used in practice. That is, one might begin by calculating the mean convergence rate for a smallest possible network of a given type following a method similar to ours used to derive Eqs. (16)–(25), then generalizing to arbitrary N through the use of energy or information optimization to find the unknown coefficients in an expression analogous to Eq. (26). Such precise specifications for learning performance are essential as hardwire versions of neural networks are implemented in various mission-critical applications. For instance, it has been shown that the perceptron learning rule can be used as a "local learning rule" in Hopfield-like associative memory networks [26]. In addition, bottom-up self-assembly of molecular nanowires [27] holds promise as technologies for which neural network paradigms could be applied to advantage. Indeed, it has been shown that agglomerations of conducting particles self-assemble, form electrical connections, and exhibit Hebbian learning through the principle of minimum resistance [28–30]. Hence higher precision insights into perceptron learning rates—gained by analyzing the geometric nature of their attractors—may have numerous practical applications.

ACKNOWLEDGMENTS

The authors acknowledge fruitful discussions with B. Reznick, G. Foster, and P. Fleck. This research was supported by National Science Foundation Grant No. NSF DMS 03–25939 ITR and NSF DGE 03–38215.

APPENDIX A: EQUIVALENT DYNAMICS

We prove that reflected weight trajectories are close approximations to the original trajectories. This is true for trajectories reflected through the line $w_1 = w_2$, which is the content of A.1, and for trajectories reflected through the origin, as shown by A.2. These two statements, combined with the definition of an average trajectory, yield A.3, which shows that the appropriately reflected average dynamics coincides for antisymmetric and opposite rules. Thus Statement A.3 gives theoretical justification for only considering 5 of the possible 14 learnable rules in our analysis, since other members of the respective classes have equivalent dynamics.

A.1 (antisymmetric rules). Let k and $\tilde{k}$ be antisymmetric rules, then if

$$\begin{pmatrix} w_{1,0}^{k} \\ w_{2,0}^{k} \end{pmatrix} = \begin{pmatrix} w_{2,0}^{\tilde{k}} \\ w_{1,0}^{\tilde{k}} \end{pmatrix}, \tag{A1}$$

$$\begin{pmatrix} w_{1,4m}^{k} \\ w_{2,4m}^{k} \end{pmatrix} = \begin{pmatrix} w_{2,4m}^{\tilde{k}} \\ w_{1,4m}^{\tilde{k}} \end{pmatrix}, \quad m = 0, 1, \ldots \;. \tag{A2}$$

This statement says that if the initial conditions of the weights are antisymmetric (reflected through the line $w_1 = w_2$), then the weight trajectories of corresponding antisymmetric rules will be antisymmetric (reflected through $w_1 = w_2$).

Proof of statement A.1. We give a proof by induction. Note that since the statement is true for $m=0$ by assumption, it suffices to prove only the induction step. Thus we suppose the statement is true for some m. Then we show that the statement is true for $m+1$, so that by the induction hypothesis, the statement will be proven. Applying Eqs. (1) and (2) repeatedly we find for rule k:

$$\begin{pmatrix} w_{1,4m+1}^{k} \\ w_{2,4m+1}^{k} \end{pmatrix} = \begin{pmatrix} w_{1,4m}^{k} \\ w_{2,4m}^{k} \end{pmatrix}, \tag{A3}$$

$$\begin{pmatrix} w_{1,4m+2}^{k} \\ w_{2,4m+2}^{k} \end{pmatrix} = \begin{pmatrix} w_{1,4m}^{k} \\ w_{2,4m}^{k} \end{pmatrix} + a[d_1^k - \Theta(w_{2,4m}^k - t)]\begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{A4}$$

$$= \begin{pmatrix} w_{1,4m}^{k} \\ w_{2,4m}^{k} + a[d_1^k - \Theta(w_{2,4m}^k - t)] \end{pmatrix}, \tag{A5}$$

$$\begin{pmatrix} w_{1,4m+3}^{k} \\ w_{2,4m+3}^{k} \end{pmatrix} = \begin{pmatrix} w_{1,4m+2}^{k} \\ w_{2,4m+2}^{k} \end{pmatrix} + a[d_2^k - \Theta(w_{2,4m+2}^k - t)]\begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{A6}$$

$$= \begin{pmatrix} w_{1,4m}^{k} + a[d_2^k - \Theta(w_{1,4m}^k - t)] \\ w_{2,4m}^{k} + a[d_1^k - \Theta(w_{2,4m}^k - t)] \end{pmatrix}, \tag{A7}$$

$$\begin{pmatrix} w_{1,4m+4}^{k} \\ w_{2,4m+4}^{k} \end{pmatrix} = \begin{pmatrix} w_{1,4m+3}^{k} + a[d_3^k - \Theta(w_{1,4m+3}^k + w_{2,4m+3}^k - t)] \\ w_{2,4m+3}^{k} + a[d_3^k - \Theta(w_{1,4m+3}^k + w_{2,4m+3}^k - t)] \end{pmatrix}. \tag{A8}$$

For rule $\tilde{k}$ we have by an analogous series of calculations,

$$\begin{pmatrix} w_{1,4m+3}^{\tilde{k}} \\ w_{2,4m+3}^{\tilde{k}} \end{pmatrix} = \begin{pmatrix} w_{2,4m}^{k} + a[d_1^k - \Theta(w_{2,4m}^k - t)] \\ w_{1,4m}^{k} + a[d_2^k - \Theta(w_{1,4m}^k - t)] \end{pmatrix} \tag{A9}$$

$$= \begin{pmatrix} w_{2,4m+3}^{k} \\ w_{1,4m+3}^{k} \end{pmatrix}, \tag{A10}$$

where use was made of the antisymmetry of the initial conditions and of the rules k and $\tilde{k}$, and Eq. (A7). Then, for $4(m+1) = 4m+4$, we have, again by antisymmetry of the rules:

$$\begin{pmatrix} w_{1,4m+4}^{\tilde{k}} \\ w_{2,4m+4}^{\tilde{k}} \end{pmatrix} = \begin{pmatrix} w_{2,4m+3}^{k} + a[d_3^{\tilde{k}} - \Theta(w_{2,4m+3}^k + w_{1,4m+3}^k - t)] \\ w_{1,4m+3}^{k} + a[d_3^{\tilde{k}} - \Theta(w_{2,4m+3}^k + w_{1,4m+3}^k - t)] \end{pmatrix} \tag{A11}$$

$$= \begin{pmatrix} w_{2,4m+4}^{k} \\ w_{1,4m+4}^{k} \end{pmatrix}, \tag{A12}$$

which verifies Eq. (A2) by the induction hypothesis.

A.2 (opposite rules). Let k and $\tilde{k}$ be opposite rules, then if

$$\begin{pmatrix} w_{1,0}^{k} \\ w_{2,0}^{k} \end{pmatrix} = -\begin{pmatrix} w_{1,0}^{\tilde{k}} \\ w_{2,0}^{\tilde{k}} \end{pmatrix}, \tag{A13}$$

$$\begin{pmatrix} w_{1,4m}^{k} \\ w_{2,4m}^{k} \end{pmatrix} = -\begin{pmatrix} w_{1,4m}^{\tilde{k}} \\ w_{2,4m}^{\tilde{k}} \end{pmatrix}, \quad m = 0, 1, \ldots \;. \tag{A14}$$

This statement says that if the initial conditions of the weights for two rules are opposite (of opposite sign), then the weight trajectories of corresponding opposite rules will be opposite (reflected through the origin).

Proof of statement A.2. We again give a proof by induction. As for A.1, the statement is true for $m=0$ by assumption, so it suffices to prove only the induction step. Thus suppose the statement is true for some m. Applying Eqs. (1) and (2) repeatedly we find for rule k Eqs. (A7) and (A8). For rule $\tilde{k}$ we have

$$\begin{pmatrix} w^{\tilde{k}}_{1,4m+1} \\ w^{\tilde{k}}_{2,4m+1} \end{pmatrix} = -\begin{pmatrix} w^{k}_{1,4m} \\ w^{k}_{2,4m} \end{pmatrix}, \tag{A15}$$

$$\begin{pmatrix} w^{\tilde{k}}_{1,4m+2} \\ w^{\tilde{k}}_{2,4m+2} \end{pmatrix} = -\begin{pmatrix} w^{k}_{1,4m} \\ w^{k}_{2,4m} \end{pmatrix} + a[\tilde{d}^{k}_{1} - \Theta(-w^{k}_{2,4m}+t)]\begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{A16}$$

$$= \begin{pmatrix} -w^{k}_{1,4m} \\ -w^{k}_{2,4m} + a[\tilde{d}^{k}_{1} - \Theta(-w^{k}_{2,4m}+t)] \end{pmatrix}, \tag{A17}$$

$$\begin{pmatrix} w^{\tilde{k}}_{1,4m+3} \\ w^{\tilde{k}}_{2,4m+3} \end{pmatrix} = \begin{pmatrix} w^{\tilde{k}}_{1,4m+2} \\ w^{\tilde{k}}_{2,4m+2} \end{pmatrix} + a[\tilde{d}^{k}_{2} - \Theta(w^{\tilde{k}}_{1,4m+2}+t)]\begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{A18}$$

$$= \begin{pmatrix} w^{\tilde{k}}_{1,4m+2} + a[\tilde{d}^{k}_{2} - \Theta(w^{\tilde{k}}_{1,4m+2}+t)] \\ w^{\tilde{k}}_{2,4m+2} \end{pmatrix} \tag{A19}$$

$$= \begin{pmatrix} -w^{k}_{1,4m} + a[\tilde{d}^{k}_{2} - \Theta(-w^{k}_{1,4m}+t)] \\ -w^{k}_{2,4m} + a[\tilde{d}^{k}_{1} - \Theta(-w^{k}_{2,4m}+t)] \end{pmatrix} \tag{A20}$$

$$= \begin{pmatrix} -w^{k}_{1,4m} + a[\tilde{d}^{k}_{2} - \tilde{\Theta}(w^{k}_{1,4m}-t)] \\ -w^{k}_{2,4m} + a[\tilde{d}^{k}_{1} - \tilde{\Theta}(w^{k}_{2,4m}-t)] \end{pmatrix} \tag{A21}$$

$$= -\begin{pmatrix} w^{k}_{1,4m} + a[d^{k}_{2} - \Theta(w^{k}_{1,4m}-t)] \\ w^{k}_{2,4m} + a[d^{k}_{1} - \Theta(w^{k}_{2,4m}-t)] \end{pmatrix}, \tag{A22}$$

thus by Eq. (A7),

$$\begin{pmatrix} w^{k}_{1,4m+3} \\ w^{k}_{2,4m+3} \end{pmatrix} = -\begin{pmatrix} w^{\tilde{k}}_{1,4m+3} \\ w^{\tilde{k}}_{2,4m+3} \end{pmatrix}. \tag{A23}$$

When $n=4(m+1)=4m+4$,

$$\begin{pmatrix} w^{\tilde{k}}_{1,4m+4} \\ w^{\tilde{k}}_{2,4m+4} \end{pmatrix} = \begin{pmatrix} -w^{\tilde{k}}_{1,4m+3} + a[\tilde{d}^{k}_{3} - \Theta(-w^{\tilde{k}}_{2,4m+3} - w^{\tilde{k}}_{1,4m+3}+t)] \\ -w^{k}_{2,4m+3} + a[\tilde{d}^{k}_{3} - \Theta(-w^{\tilde{k}}_{2,4m+3} - w^{\tilde{k}}_{1,4m+3}+t)] \end{pmatrix} \tag{A24}$$

$$= \begin{pmatrix} -w^{k}_{1,4m+3} + a[\tilde{d}^{k}_{3} - \Theta(-w^{k}_{2,4m+3} - w^{k}_{1,4m+3}+t)] \\ -w^{k}_{2,4m+3} + a[\tilde{d}^{k}_{3} - \Theta(-w^{k}_{2,4m+3} - w^{k}_{1,4m+3}+t)] \end{pmatrix} \tag{A25}$$

$$= \begin{pmatrix} -w^{k}_{1,4m+3} + a[\tilde{d}^{k}_{3} - \tilde{\Theta}(w^{k}_{2,4m+3} + w^{k}_{1,4m+3}-t)] \\ -w^{k}_{2,4m+3} + a[\tilde{d}^{k}_{3} - \tilde{\Theta}(w^{k}_{2,4m+3} + w^{k}_{1,4m+3}-t)] \end{pmatrix} \tag{A26}$$

$$= -\begin{pmatrix} w^{k}_{1,4m+4} \\ w^{k}_{2,4m+4} \end{pmatrix}, \tag{A27}$$

using Eqs. (A23) and (A3).

A.3 (average dynamics). If k and $\tilde{k}$ are antisymmetric or opposite rules, then the average dynamics of k and $\tilde{k}$ are equivalent. That is,

$$\begin{pmatrix} \langle w^{k}_{1,n} \rangle \\ \langle w^{k}_{2,n} \rangle \end{pmatrix} = \begin{pmatrix} \langle w^{\tilde{k}}_{2,n} \rangle \\ \langle w^{\tilde{k}}_{1,n} \rangle \end{pmatrix}, \quad n=0,1,\ldots, \tag{A28}$$

for antisymmetric rules, or

$$\begin{pmatrix} \langle w^{k}_{1,n} \rangle \\ \langle w^{k}_{2,n} \rangle \end{pmatrix} = -\begin{pmatrix} \langle w^{\tilde{k}}_{1,n} \rangle \\ \langle w^{\tilde{k}}_{2,n} \rangle \end{pmatrix}, \quad n=0,1,\ldots, \tag{A29}$$

for opposite rules, where

$$\langle w_{i,n} \rangle \equiv \frac{1}{4}\sum_{m=n}^{n+3} w_{i,m}, \quad 1 \leqslant i \leqslant N. \tag{A30}$$

Proof of statement A.3. We again give a proof by induction. Suppose first that k and $\tilde{k}$ are antisymmetric rules. Then from the induction assumption

$$\begin{pmatrix} \langle w^{\tilde{k}}_{1,n} \rangle \\ \langle w^{\tilde{k}}_{2,n} \rangle \end{pmatrix} = \begin{pmatrix} \langle w^{k}_{2,n} \rangle \\ \langle w^{k}_{1,n} \rangle \end{pmatrix}, \tag{A31}$$

we have

$$\begin{pmatrix} \langle w^{\tilde{k}}_{1,n+1} \rangle \\ \langle w^{\tilde{k}}_{2,n+1} \rangle \end{pmatrix} = \begin{pmatrix} \frac{1}{4}\sum_{m=n+1}^{n+4} w^{\tilde{k}}_{1,m} \\ \frac{1}{4}\sum_{m=n+1}^{n+4} w^{\tilde{k}}_{2,m} \end{pmatrix} \tag{A32}$$

$$= \begin{pmatrix} \langle w^{\tilde{k}}_{1,n} \rangle \\ \langle w^{\tilde{k}}_{2,n} \rangle \end{pmatrix} + \frac{1}{4}\begin{pmatrix} w^{\tilde{k}}_{1,n+4} - w^{\tilde{k}}_{1,n} \\ w^{\tilde{k}}_{2,n+4} - w^{\tilde{k}}_{2,n} \end{pmatrix} \tag{A33}$$

$$= \begin{pmatrix} \langle w^{k}_{2,n} \rangle \\ \langle w^{k}_{1,n} \rangle \end{pmatrix} + \tilde{\Delta}. \tag{A34}$$

Then, since

$$\begin{pmatrix}\langle w^k_{2,n+1}\rangle\\ \langle w^k_{1,n+1}\rangle\end{pmatrix}=\begin{pmatrix}\langle w^k_{2,n}\rangle\\ \langle w^k_{1,n}\rangle\end{pmatrix}+\frac{1}{4}\begin{pmatrix}w^k_{2,n+4}-w^k_{2,n}\\ w^k_{1,n+4}-w^k_{1,n}\end{pmatrix} \quad (A35)$$

$$=\begin{pmatrix}\langle w^k_{2,n}\rangle\\ \langle w^k_{1,n}\rangle\end{pmatrix}+\Delta, \quad (A36)$$

it suffices to verify that

$$4(\tilde{\Delta}-\Delta)=\begin{pmatrix}w^{\tilde{k}}_{1,n+4}-w^{\tilde{k}}_{1,n}-w^k_{2,n+4}+w^k_{2,n}\\ w^{\tilde{k}}_{2,n+4}-w^{\tilde{k}}_{2,n}-w^k_{1,n+4}+w^k_{1,n}\end{pmatrix} \quad (A37)$$

$$=0. \quad (A38)$$

Proceeding case by case for the values of n, we have first the case $n=4m$, which yields

$$4(\tilde{\Delta}-\Delta)=\begin{pmatrix}w^{\tilde{k}}_{1,4(m+1)}-w^k_{2,4(m+1)}-(w^{\tilde{k}}_{1,4m}-w^k_{2,4m})\\ w^{\tilde{k}}_{2,4(m+1)}-w^k_{1,4(m+1)}-(w^{\tilde{k}}_{2,4m}-w^k_{1,4m})\end{pmatrix} \quad (A39)$$

$$=0, \quad (A40)$$

by Appendix statement A.1. When $n=4m+1$, $4(\tilde{\Delta}-\Delta)=0$ by Eq. (A3). If $n=4m+2$, then

$$4(\tilde{\Delta}-\Delta)_1=w^{\tilde{k}}_{1,4m+6}-w^{\tilde{k}}_{2,4m+2}-w^k_{2,4m+6}+w^k_{2,4m+2} \quad (A41)$$

$$=w^k_{2,4m+4}-w^k_{2,4m+2}-w^k_{2,4m+4}+w^k_{2,4m+2} \quad (A42)$$

$$=0, \quad (A43)$$

and

$$\begin{aligned}4(\tilde{\Delta}-\Delta)_2=&\,w^k_{1,4m+4}+a[d^k_2-\theta(w^k_{1,4m+4}-t)]-w^k_{1,4m}\\ &-a[d^k_2-\theta(w^k_{1,4m}-t)]-w^k_{1,4m+4}+w^k_{1,4m}\end{aligned} \quad (A44)$$

$$=a[\theta(w^k_{1,4m}-t)-\theta(w^k_{1,4m+4}-t)] \quad (A45)$$

$$\cong 0, \quad (A46)$$

where Eq. (A5) was used. (The second component is bounded by $|a|\ll 1$ when nonzero, and is only nonzero when the weight trajectory crosses some regional boundary.) For $n=4m+3$, $4(\tilde{\Delta}-\Delta)=0$ by Eq. (A10). Thus if Eq. (A28) is true for n, it is true for $n+1$. It is true in particular for $n=0$, since this is the case $n=4m$. Hence by the induction hypothesis, Eq. (A28) is verified for all n.

Now suppose that k and $\tilde{k}$ are opposite rules. Assume the statement is true for n, so that

$$\begin{pmatrix}\langle w^{\tilde{k}}_{1,n}\rangle\\ \langle w^{\tilde{k}}_{2,n}\rangle\end{pmatrix}=-\begin{pmatrix}\langle w^k_{1,n}\rangle\\ \langle w^k_{2,n}\rangle\end{pmatrix}. \quad (A47)$$

Then

$$\begin{pmatrix}\langle w^{\tilde{k}}_{1,n+1}\rangle\\ \langle w^{\tilde{k}}_{2,n+1}\rangle\end{pmatrix}=-\begin{pmatrix}\langle w^k_{1,n}\rangle\\ \langle w^k_{2,n}\rangle\end{pmatrix}+\tilde{\Delta}, \quad (A48)$$

and

$$-\begin{pmatrix}\langle w^k_{1,n+1}\rangle\\ \langle w^k_{2,n+1}\rangle\end{pmatrix}=-\begin{pmatrix}\langle w^k_{1,n}\rangle\\ \langle w^k_{2,n}\rangle\end{pmatrix}+\Delta, \quad (A49)$$

where

$$\tilde{\Delta}=\frac{1}{4}\begin{pmatrix}w^{\tilde{k}}_{1,n+4}-w^{\tilde{k}}_{1,n}\\ w^{\tilde{k}}_{2,n+4}-w^{\tilde{k}}_{2,n}\end{pmatrix}, \quad (A50)$$

and

$$\Delta=\frac{1}{4}\begin{pmatrix}w^k_{1,n}-w^k_{1,n+4}\\ w^k_{2,n}-w^k_{2,n+4}\end{pmatrix}. \quad (A51)$$

We then proceed as for the antisymmetric rules by considering cases on n. When $n=4m$, $4m+1$, or $4m+3$, we find $\tilde{\Delta}-\Delta=0$ using Appendix statement A.2, Eq. (A15), or Eq. (A23), respectively, by exactly analogous calculations as for the antisymmetric rules. If $n=4m+2$, then by Eqs. (A5) and (A17) we get

$$4(\tilde{\Delta}-\Delta)_1=w^{\tilde{k}}_{1,4m+6}-w^{\tilde{k}}_{1,4m+2}-w^k_{1,4m+2}+w^k_{1,4m+6} \quad (A52)$$

$$=-w^k_{1,4m+4}+w^k_{1,4m}-w^k_{1,4m}+w^k_{1,4m+4} \quad (A53)$$

$$=0, \quad (A54)$$

and

$$\begin{aligned}4(\tilde{\Delta}-\Delta)_2=&-w^k_{2,4m+4}+a[\tilde{d}^k_1-\theta(-w^k_{2,4m+4}+t)]+w^k_{2,4m}\\ &-a[\tilde{d}^k_1-\theta(-w^k_{2,4m}+t)]-w^k_{2,4m}\\ &-a[d^k_1-\theta(w^k_{2,4m}-t)]+w^k_{2,4m+4}\end{aligned}$$

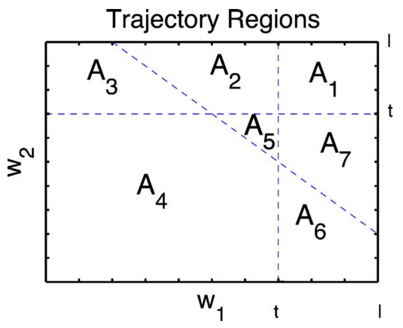

FIG. 4. (Color online) Regions A_1–A_7 used for calculating Eq. (20) from Eqs. (10) and (12). The dashed lines corresponding to $w_1=t$, $w_2=t$, and $w_2=-w_1+t$ partition the initial weight vector space into the seven regions A_1–A_7.

TABLE III. Details utilized to determine all possible trajectories for weight vectors initially in the different regions A_1–A_7. The condition violated was used to select the weight change, which in turn gave the slope of the trajectory segment. By determining the distance d_R of the complete trajectory from arbitrary starting location to the attractor A_6, and multiplying by the rate r_R for each different segment, the integral of Eq. (B1) was written down.

Region	Conditions violated	Weight changes	Slope	r_R
A_1	$w_2<t,\ w_2<-w_1+t$	$a\langle 0,-1\rangle+a\langle -1,-1\rangle$	2	$\frac{2}{a\sqrt{5}}$
A_2	$w_1>t,\ w_2<t,\ w_2<-w_1+t$	$a\langle 1,0\rangle+a\langle 0,-1\rangle+\langle -1,-1\rangle$	∞	$\frac{3}{2a}$
A_3	$w_1>t,\ w_2<t$	$a\langle 1,0\rangle+a\langle 0,-1\rangle$	-1	$\frac{\sqrt{2}}{a}$
A_4	$w_1>t$	$a\langle 1,0\rangle$	0	$\frac{1}{a}$
A_5	$w_1>t,\ w_2<-w_1+t$	$a\langle 1,0\rangle+a\langle -1,-1\rangle$	∞	$\frac{2}{a}$
A_7	$w_2<-w_1+t$	$a\langle -1,-1\rangle$	1	$\frac{1}{a\sqrt{2}}$

$$\begin{aligned}&+a[d_1^k-\theta(w^k_{2,4m+4}-t)]\\
&=a[\tilde{d}_1^k-\tilde{\theta}(w^k_{2,4m+4}-t)]-a[\tilde{d}_1^k-\tilde{\theta}(w^k_{2,4m}-t)]\\
&\quad-a[d_1^k-\theta(w^k_{2,4m}-t)]+a[d_1^k-\theta(w^k_{2,4m+4}-t)]\\
&=0. \qquad \text{(A55)}\end{aligned}$$

Since $n=0$ is included in the case $n=4m$, this completes the induction for the opposite rules and thus the proof of statement A.3.

APPENDIX B: EXAMPLE CALCULATION

In this appendix, we sketch the calculation of Eq. (20), using Eqs. (10) and (12). In Fig. 4 is shown the seven different regions to consider. If the initial weight value is in region A_6, then all the inequalities in the third line of Table I are satisfied, so that the weight vector is already in the attractor. If, however, the weight vector is in one of the other regions, then it will follow its trajectory to the attractor (A_6) as shown in Fig. 2(c). Thus in order to find the integral in Eq. (10), the length of each possible trajectory is calculated. First each possible trajectory is determined to yield Eq. (12). Then Eq. (12) is used to compute the component distances d_R in Eq. (14), in each segment of the trajectory. Finally, the rates r_R in Eq. (14), which give the number of steps per unit distance traveled by the weight vector, are calculated. In Table III we give the details which were used to write down the integral for Eq. (10). Putting these details together, the integral for Eq. (10) was found to be

$$\begin{aligned}\mu_3(a,l,t)=\frac{1}{4al^2}\Bigg\{&\int_0^t\int_{t-y}^t[2(x+y-t)+3(t-x)]dxdy+\int_0^t\int_t^{t+y}[(x-t)+2(t+y-x)]dxdy+\int_0^t\int_{t+y}^l(x/2-t/2+y/2)dxdy\\
&+\int_{t-l}^0\int_{t-y}^l(x/2-t/2+y/2)dxdy+\int_0^t\int_{-l}^{t-y}[(t-y-x)+3y]dxdy+\int_{-l}^0\int_{-l}^t(t-x)dxdy\\
&+\int_t^l\int_{-l}^{t-y}[2(y-t)+(t-x-y)+3t]dxdy+\int_t^l\int_{t-y}^t[3/2(x+y-t)+(-2x)+3t]dxdy\\
&+\int_t^l\int_0^t[3/2(y-t)+2x+3(t-x)]dxdy+\int_t^l\int_{3t/2+y/2}^{t/2+y/2}[2(x-t)+4/3(t-2x+y)+2t]dxdy\\
&+\int_t^l\int_{t/2+y/2}^{3t/2+y/2}[(y-t)+(-t/2+x-y/2)+2(3t/2-x+y/2)]dxdy+\int_t^l\int_{3t/2+y/2}^l[(y-t)+(t/4+x/2-y/4)]dxdy\Bigg\}.\end{aligned}\qquad\text{(B1)}$$

Integrating and simplifying this expression with MATHEMATICA then yields Eq. (20).

[1] S. Miyoshi, K. Hara, and M. Okada, Phys. Rev. E **71**, 036116 (2005).
[2] C. Monterola and M. Zapotocky, Phys. Rev. E **71**, 036134 (2005).
[3] W. Senn and S. Fusi, Phys. Rev. E **71**, 061907 (2005).
[4] T. L. H. Watkin, A. Rau, and M. Biehl, Rev. Mod. Phys. **65**, 499 (1993).
[5] D. Bolle, I. Perez Castillo, and G. M. Shim, Phys. Rev. E **67**, 036113 (2003).
[6] M. Rosen-Zvi, A. Engel, and I. Kanter, Phys. Rev. E **66**, 036138 (2002).
[7] R. Urbanczik, Phys. Rev. E **68**, 016106 (2003).
[8] A. Freking, W. Kinzel, and I. Kanter, Phys. Rev. E **65**, 050903(R) (2002).
[9] A. Priel and I. Kanter, Europhys. Lett. **51**, 230 (2000).
[10] A. Priel and I. Kanter, Phys. Rev. E **59**, 3368 (1999).
[11] M. Schroder and W. Kinzel, J. Phys. A **31**, 2967 (1998).
[12] H. Suyari and I. Matsuba, Phys. Rev. E **60**, 4576 (1999).
[13] L. Diambra and J. J. Fernandez, Phys. Rev. E **64**, 046106 (2001).
[14] M. Heo, S. Kim, E. J. Moon, M. Cheon, K. Chung, and I. Chang, Phys. Rev. E **72**, 011906 (2005).
[15] B. Widrow and M. A. Lehr, Proc. IEEE **78**, 1415 (1990).
[16] W. Wu and Z. Shao, Appl. Math. Lett. **16**, 999 (2003).
[17] H. O. Carmesin, Phys. Rev. E **50**, 622 (1994).
[18] U. Bhattacharya and S. K. Parui, Pattern Recogn. Lett. **16**, 491 (1995).
[19] T. Heskes and W. Wiegerinck, IEEE Trans. Neural Netw. **7**, 919 (1996).
[20] L. E. Arsenault and A. W. Hubler, Phys. Rev. E **51**, 3561 (1995).
[21] C. Wargitsch and A. Hubler, Phys. Rev. E **51**, 1508 (1995).
[22] B. A. Huberman, R. M. Lukose, and T. Hogg, Science **275**, 51 (1997).
[23] A. L. Blum and R. L. Rivest, Neural Networks **5**, 117 (1992).
[24] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky, Nature (London) **400**, 133 (1999).
[25] G. Polya, *Mathematik und Plausibles Schliessen* (Birkhaeuser, Basel, Switzerland, 1962).
[26] S. Diederich and M. Opper, Phys. Rev. Lett. **58**, 949 (1987).
[27] T. Akutagawa, O. Takanori, H. Tatsuo, N. Takayoshi, C. Christensen, and J. Becher, Proc. Natl. Acad. Sci. U.S.A. **99**, 5028 (2002).
[28] M. Sperl, A. Chang, N. Weber, and A. Hubler, Phys. Rev. E **59**, 3165 (1999).
[29] M. Dueweke, U. Dierker, and A. Hubler, Phys. Rev. E **54**, 496 (1996).
[30] J. K. Jun and A. W. Hubler, Proc. Natl. Acad. Sci. U.S.A. **102**, 536 (2005).