On-line training of a continually adapting adaline-like network

Morris W. Hirsch*1
Department of Mathematics, University of California at Berkeley, Berkeley, CA 94720-3840, USA
Received 6 April 1995; accepted 28 January 1997

Abstract

A rigorous mathematical analysis is presented of an adaline-like network operating in continuous time with spatially continuous inputs and outputs. Weights adapt continually, whether or not a training signal is present. It is shown that consistent input–output pairs can be learned perfectly provided every pattern is repeated at least once in every \( N \) successive inputs, and the input patterns are nearly orthogonal, depending on \( N \).

Keywords: Attractors; Dynamics; Neural networks; On-line training; Supervised learning

0. Introduction

In the vast majority of nets studied or used for pattern classification, training the net is clearly separated from running it; training and running are implemented by different algorithms. There are obvious practical advantages in this procedure, and alternatives are not well understood.

There may, however, be practical situations where an incoming stream of data must be used to further train the net at the same time that it is classified by the net. Biologically, on-line training seems to be the rule. We learn while doing – we learn by doing, and if we don’t do, we may forget.

For these reasons it is of interest to investigate on-line training algorithms, in which testing reinforces learning, lack of testing can lead to forgetting, and retraining at any time is possible.

* Tel.: 1 510 642 4318; fax: 510 642 8204; e-mail: hirsch@math.berkeley.edu.
1 Partially supported by a grant from the National Science Foundation.
It seems obvious that in any such training system there are constraints and relations between quantities such as the following:

(a) the number of patterns a system can learn to classify;
(b) the length of time a system can remember the classification;
(c) the ease of learning new classifications;
(d) the mutual separation of the patterns;
(e) the length of time each pattern is presented;
(f) the frequency with which a pattern is repeated, for training or testing.

It is not clear how to investigate such quantities, or even what some of them mean, for training methods based on error reduction, such as back propagation, where training the net and running (or testing) it are separate processes.

Here we will look at a class of networks for supervised learning of pattern classification. They are run by differential equations, although this is not essential. Their most significant feature is the training method: the only difference between training and testing is the presence of a training signal. The memory vector is *always* adapting, during both training and testing.

We examine an on-line training scheme for a simple adaline-like system, whose purpose is to sort a given set of input pattern vectors into two classes. The feature of interest is that weights and outputs continually evolve according to differential equations having the input patterns as parameters. The only difference between training the net and running it, is the presence of a training signal. Under suitable constraints on the input patterns and learning parameters, the system is guaranteed to learn the correct output for each input pattern on which it has been trained at least once, and for which no inconsistent training signals have been given.

While the specific system presented here has the signal virtue of permitting mathematically rigorous analysis, it is doubtful whether it has any practical application, since the set of classifications it can provably learn requires input patterns to be nearly orthogonal, depending on the number of patterns. On-line learning systems, however, ought to be useful in contexts where batch processing is not possible or not desirable, as when input patterns are not known in advance. A very interesting avenue for research, touched on in Section 5, is the extent to which on-line schemes can adapt to nonstationary input streams. The system described here has this capability, in the limited context of nearly orthogonal patterns, although the proof is not included here.

The key constraints are given in terms of two numbers \( \varepsilon, N \). Here \( \varepsilon > 0 \) denotes an upper bound for the absolute value of the inner (dot) product between distinct input patterns. The integer \( N \geq 1 \) depends on the sequence \( \{ P_k \} \) of input patterns presented to the network: It is assumed that each pattern is presented at least once in every \( N + 1 \) successive patterns. Thus the number of patterns is \( \leq N + 1 \), but might be much smaller.

Input patterns are chosen from a finite set \( \mathcal{P} = \{ P^* \} \) of vectors in some Euclidean space \( \mathbf{R}^d \); for convenience they are taken to be unit vectors. At each of a sequence of times \( t_k \), a pattern \( P_k \) and a training signal \( t_k \in \{ -1, 0, +1 \} \) are simultaneously input; these are held constant while the weight vector \( M(t) \in \mathbf{R}^d \) and the output \( y(t) \in [-1, 1] \) evolve according to differential equations. The numbers \( t_k - t_{k-1} \) are assumed to exceed some lower bound \( T_* \) – an important parameter – but are otherwise arbitrary.
Certain patterns $P^a$ — not necessarily all — are associated with definite training signals $I^a \in \{-1, 1\}$. This means that whenever $P_k = P^a$, either $I_k = 0$ (no training) or $I_k = I^a$, and the latter happens at least once. Such a pair $(P^a, I^a)$ is called eventually consistent. The goal is to have the net learn to output $I^a$ whenever $P^a$ is input.

Our results, Theorems 4.2 and 4.3, show that if $\varepsilon N$ is sufficiently small and $T_\ast$ is sufficiently large, then this kind of learning is possible. Moreover, a priori bounds on the weight vector $M(t)$ can be enforced.

Since the network runs by attractor dynamics, the network is able to generalize from the given set of input patterns to nearby patterns (although we do not prove this here): The results continue to hold where patterns are allowed to vary within sufficiently small open sets.

Sections 1 and 2 describe the basic operation of the network, including its dynamics for a single pattern — a simple planar vector field. Section 3 shows how the conflict between patterns is resolved for almost orthogonal patterns. The main theorems are stated in Section 4; proofs are postponed to Section 6. Section 5 discusses the results.

0.1. Comparison with other approaches

On-line algorithms which continually adapt have been discussed many times, especially in the literature on adaptive algorithms in signal processing and control on linearly parameterized classes. See, e.g. [3–5, 12]. For a survey of learning linear subspaces, see [2].


The assumption of near orthogonality and frequent presentation of input patterns is analogous to the following condition of persistent excitation used in [4, 5, 12]: For the sequence of successive input patterns $P_j \in \mathbb{R}^d$, $j = 1, 2, \ldots$, it is required that there exist constant $\beta > \alpha > 0$ and an integer $S > d$ such that for all $j$, the eigenvalues of the symmetric matrix $\frac{1}{S} \sum_{j=1}^{S} P_j P_j^T$ lie in the interval $[\alpha, \beta]$; here $P_j^T$ denotes the transpose of column vector $P_j$. This is easily seen to be implied by Hypothesis 4.1 if we take $\mathbb{R}^d$ to be the linear space spanned by the input patterns, and we take $\varepsilon$ sufficiently small relative to $N$.

There is a strong resemblance between the scheme presented here and adaline learning [20]; and between network dynamics discussed here and the dynamics of continuous-time additive nets studied by Hopfield, Grossberg, and many others. It is well known that single-layer perceptrons can learn orthogonal patterns perfectly, as can Hopfield networks [11]. Perceptrons, however, can also learn any linearly separable set of patterns; a similar result for the scheme in this paper has proved elusive.

2 I am indebted to the anonymous reviewers, and to Professor R.C. Williamson of The Australian National University, for much of the following discussion, as well as for a careful reading and helpful suggestions.
On the other hand, the results proved here extend easily to allow slowly varying sets of input patterns: Instead of drawing each input from a fixed set \( \mathcal{P} \) of patterns, we can allow the input at each time \( k \) to be taken from a set \( \mathcal{P}_k \) of patterns sufficiently close to the previous set \( \mathcal{P}_{k-1} \), and the net will learn in the same way. This allows the possibility of changing the set of input–output pairs to a totally different set, while preserving perfect network behavior – provided the change is done slowly enough, with sufficient orthogonality.

In many approaches it is assumed that the sequence of patterns satisfies strong probabilistic constraints, and the mathematical conclusions hold with probability one. In the present work there are no probabilistic assumptions, but the hypothesis that each pattern appears at least once in every successive \( N \) patterns is not satisfied in most stochastic contexts.

In many learning algorithms, it is necessary to run through the full set of input patterns many times before learning is achieved. Under the present scheme, learning is achieved in a single pass through the pattern set. Moreover, the classification of any of the patterns can be changed at any time by simply presenting the relevant patterns once with the new teaching signal, whereupon the new classifications will be learned.

In a related approach [10], I show that similar on-line training schemes can teach a system to associate, to each input pattern from a nearly orthogonal set, a specified attractor \( A \) in a given vector field \( F \), provided the basin of \( A \) contains a known convex set. In this sense on-line training can teach a system to respond to inputs with different dynamical output, represented by various attractors. The present paper is a very special case: the vector field is \( \frac{dy}{dt} = g(y) \), where \( g \) is the sigmoidal transfer function used in Eq. (1), and the attractors are the two stable equilibria \( \pm 1 \).

Reviewers have remarked on resemblances between the behavior of the network described here and those investigated by Almeida [1] and Pineda [16].

1. A feature detector

The purpose of the continuous-time system described below is to classify a finite set of input vectors \( P \in \mathbb{R}^d \) into two arbitrary classes by means of a continuous output signal \( y \in \mathbb{R} \) which, upon presentation of a pattern, limits at \( \pm 1 \). A training signal \( I \in \mathbb{R} \) can be given at any time; \( I = 0 \) means the system is being used for classification, not training. It is not necessary that a “correct” output be associated to every input pattern.

The system compares the current input pattern \( P \) with a memory vector \( M(t) \in \mathbb{R}^d \) by passing the inner product \( M \cdot P \) through a sigmoidal transfer function \( g \), and computing its output \( y(t) \in \mathbb{R} \) by means of a differential equation:

\[
\frac{dy}{dt} = -y + g(M \cdot P + I). \tag{1}
\]

At the same time, the memory continually adapts according to

\[
\frac{dM}{dt} = \lambda(-M \cdot P + y)P, \tag{2}
\]
where $\lambda > 0$ is a kind of learning rate. Thus we have the $d+1$-dimensional system:

\[
\frac{dy}{dt} = -y + g(M \cdot P + I),
\]

\[
\frac{dM}{dt} = \lambda(-M \cdot P + y)P.
\]

Notice that $I$ and $P$ determine the dynamics.

It is interesting to note that System (3), (4) is equivalent to a single second order vector equation for $M$. When $P$ is a unit vector, as we shall assume, this equation is:

\[
\frac{d^2M}{dt^2} + \left[(\lambda + 1)\frac{dM}{dt} + \lambda M \cdot P\right]P = [g(M \cdot P + I)]P,
\]

We always assume:

- $P$ is a unit vector.
- $I \in \{-1, 0, 1\}$.
- The transfer function $g: \mathbb{R} \to [-1, 1]$ is a ramp function with gain $\kappa > 1$:

\[
g(s) = \begin{cases} 
-1 & \text{if } s \leq -1/\kappa, \\
\kappa s & \text{if } |s| \leq 1/\kappa, \\
+1 & \text{if } s \geq 1/\kappa.
\end{cases}
\]

Evidently $g(s)$ has fixed points $s = -1, 0, 1$ and nowhere else. It satisfies a Lipschitz condition, and is continuously differentiable except at the two points $\pm 1/\kappa$. At the fixed points the derivatives are

\[
g'(\pm 1) = 0, \quad g'(0) = \kappa.
\]

Note that under these assumptions, the function $g(s + 1)$ has a unique fixed point at $s = 1$, while $g(s - 1)$ has a unique fixed point at $s = -1$.

To analyze this system we introduce the recall variable $x(t) = M(t) \cdot P$, interpreted as the system's current classification of pattern $P$. Dotting (4) with $P$ yields the two-dimensional system:

\[
\frac{dx}{dt} = \lambda(-x + y),
\]

\[
\frac{dy}{dt} = -y + g(x + I).
\]

These equations are conveniently independent of the input pattern $P$ but the meaning of $x$ depends on $P$. When $P$ is changed, the value of $x(t)$ jumps discontinuously. Dotting (5) with $P$ yields a second order equation for $x(t)$:

\[
\frac{d^2x}{dt^2} + (\lambda + 1)\frac{dx}{dt} + x = g(x + I).
\]
This is reminiscent of the second order activation dynamics used by Freeman et al. [6–8].

2. Phase plane analysis

If \( I = 0 \) there are two (linearly) stable equilibria of System (6), (7), at \((1, 1)\) and \((-1, -1)\), and one unstable equilibrium at \((0, 0)\). If \( I \in \{-1, 1\} \), there is a unique equilibrium at \((I, I)\), and it is globally asymptotically stable.

We denote the flow of System (6), (7) by \( \Phi_t \). For each \( t \) we have a homeomorphism \( \Phi_t \) of the plane such that the solution \( z(t) = (x(t), y(t)) \) with initial value \( z(0) \) is the curve \( t \mapsto \Phi_t(z(0)) \). It is easy to see that every forward orbit is bounded. Because the vector field defined by the right hand side of System (6), (7) has negative divergence almost everywhere, \( \Phi_t \) decreases area for \( t > 0 \). It follows from the generalized Poincaré–Bendixson theorem [9] that every forward trajectory converges to an equilibrium.

Therefore we have:

**Proposition 2.1.** Fix \( P \) and \( I \) in System (3), (4) and System (6), (7). Fix \( t_0 \geq 0 \), choose initial values \( y(t_0), M(t_0) \), and the corresponding initial value \( x(t_0) = M(t_0) \cdot P \). Let \((y(t), M(t))\) be the corresponding solution of System (3), (4) and \((x(t), y(t))\) the corresponding solution of System (6), (7). Then as \( t \to \infty \) these solutions converge to respective equilibria \((y_*, M_*)\) and \((x_*, y_*)\), such that:

\[
\begin{align*}
y_* &= M_* P, \\
y_* &= g(M_* \cdot P + I) \\
x_* &= y_*.
\end{align*}
\]

Equilibrium \((x_*, y_*)\) is stable if and only if \((y_*, M_*)\) is stable, and then

\( x_* = y_* = g(y_*) \in \{-1, 1\} \).

If \( I \in \{-1, 1\} \) then the equilibria are unique for their systems, and

\[
x_* = y_* = g(y_*) = I.
\]

We can now see how to train this net on a single pattern \( P \) to produce a desired output \( J \in \{-1, 1\} \): Set \( I = J \) and run System (3), (4) (and concurrently System (6), (7)) until at \( t = T \) it is very close to the unique equilibrium. Then

\[
(x(T), y(T)) \approx (J, J).
\]

Now set \( I = 0 \) and run the systems for \( t \geq T \), starting at \( M(T), x(T) \) and \( y(T) \). The key point is that \((x(T), y(T))\) is in the basin of attraction of the stable equilibrium at \((J, J)\) for System (6), (7) with \( I = 0 \). Thus \( \lim_{t \to \infty} y(t) \to J \), as required.
It is not obvious as to what happens as the system is trained on new input patterns. It turns out to depend on the order, frequency, duration and rate of training, and on the geometry of the patterns. In the following section we investigate two patterns.

3. Training on two patterns

Simple examples show that training on a pattern $P^a$ can be undone by training on a second pattern, or even the mere presentation of a second pattern, provided it is sufficiently different.

But this cannot happen if the input patterns are close to being orthogonal. To see this, consider two patterns $P^a, P^b$, unit vectors in $\mathbb{R}^d$. Set

$$x^a(t) = M(t) \cdot P^a, \quad x^b(t) = M(t) \cdot P^b.$$  

Suppose the net has been trained during time interval $[t_0, t_1]$ on $P^a$, so that $M(t_1) \cdot P^a$, and $y(M(t_1) \cdot P^a)$ are sufficiently close to the desired output $I^a$:

$$x^a(t_1) \approx y(t_1) \approx I^a.$$

Now replace $P^a$ by $P^b$ and use training signal $I = I^a$, running System (6), (7) for $t$ in a sufficiently large interval $[t_1, t_2]$ so that

$$x^a(t_2) \approx y(t_2) \approx I^a.$$

What will happen if now at time $t_2$ we instantaneously switch $P$ to pattern $P^b$ and run System (3), (4) $t \geq t_2$, starting at the initial value $(y(t_2), M(t_2))$, and allowing either $I = 0$ or $I = I^b$? To find out we need to know $x^b(t_2)$. The relevant question is: What happened to $x^a(t)$ while $P^a$ was input? To answer this, we compute $dx^a/dt$ from (4) for $t \in [t_1, t_2]$, setting $P = P^a$:

$$\frac{dx^b}{dt} = \frac{d(M \cdot P^a)}{dt} = \frac{dM}{dt} \cdot P^b$$

$$= [\lambda \{ - M \cdot P^a + y \} P^a] \cdot P^b = \lambda \{ - M \cdot P^a + y \} (P^a \cdot P^b)$$

$$= \frac{d(M \cdot P^a)}{dt} (P^a \cdot P^b).$$

This shows that

$$\frac{dx^a}{dt} = (P^a \cdot P^b) \frac{dx^a}{dt}$$

while pattern $P^a$ is input. By integration we obtain:
Proposition 3.1. Suppose pattern \( P^a \) is input during time period \( t_1 \leq t \leq t_2 \). Then
\[
x(t_2) - x(t_1) = (P^a \cdot P^\beta) [x(t_2) - x(t_1)].
\] (10)

Corollary 3.2. If
\[
|P^a \cdot P^\beta| < \varepsilon
\] then
\[
|x(t_2) - x(t_1)| < \varepsilon |x(t_2) - x(t_1)|.
\] (11)

Since System (6), (7) is easily proved to have a global attractor, there is an a priori bound on \( |x(t_2) - x(t_1)| \). It follows that \( x(t_2) \) will be as close as desired to \( x(t_1) \), and thus to \( P^\beta \), provided \( |P^a \cdot P^\beta| \) is sufficiently small.

Even though \( x(t_2) \approx P^\beta \), we have \( y(t_2) \approx I^a \). There are two cases to consider: \( I^a = I^\beta \) and \( I^a = -I^\beta \).

If \( I^a = I^\beta \) then \( (x(t_2), y(t_2)) \approx (I^\beta, I^\beta) \), and we may assume \( (x(t_2), y(t_2)) \) is in the basin of attraction of the stable equilibrium that System (6), (7) has at \( (I^\beta, I^\beta) \). Therefore if the input pattern is changed back to \( P^\beta \) in System (3), (4), System (6), (7) will converge to \( (I^\beta, I^\beta) \) provided \( I \in \{0, I^\beta\} \). Thus in this case \( y(t) \) will converge as \( t \to \infty \) to the correct output \( I^\beta \).

What if \( I^a = -I^\beta \)? For definiteness, let \( I^a = 1, I^\beta = -1 \). Then \( (x(t_2), y(t_2)) \approx (-1, 1) \). We run System (6), (7) for \( t \geq t_2 \) with initial value \( (x(t_2), y(t_2)) \approx (-1, 1) \) and \( I \in \{0, -1\} \). We want \( y(t) \) to converge to \(-1\).

This must happen if \( I = -1 \), since then the unique equilibrium at \((-1, -1)\) is globally asymptotically stable.

Suppose \( I = 1 \). The key point is this: if \( \lambda \) is small enough, then \((-1, 1)\) is in the basin of attraction \( \mathcal{B}(-1, -1) \) of the stable equilibrium at \((-1, -1)\). To see this, first consider System (6), (7) with \( \lambda = 0 \). Evidently every forward trajectory converges on a vertical path to a point on the graph \( y = g(x) \) (under our assumption that \( I = 0 \)). It follows from continuity of solutions with respect to parameters that for sufficiently small \( \lambda \), the trajectory for \( t \geq t_2 \) of \((-1, -1)\) reaches in finite time a point which is very near \((-1, -1)\), hence in \( \mathcal{B}(-1, -1) \). This means that \((-1, 1) \in \mathcal{B}(-1, -1) \), because basins are open sets. It follows that a whole neighborhood of \((-1, 1)\) is in the basin, and we may assume \( (x(t_2), y(t_2)) \) is in such a neighborhood. Thus input pattern \( P^\beta \) makes \( y(t) \) converge to the correct output \(-1\).

4. Training on several patterns

Suppose there are many patterns, but not more than \( N + 1 \). Denote the set of patterns by \( \mathcal{P} \subset \mathbb{R}^d \). The patterns and training signals are presented in a sequence
\[
(P^k, J^k) \in \mathcal{P} \times \{-1, 0, 1\}, \quad k \in \mathbb{N}_+.
\]
where $N_+$ denotes the set of positive natural numbers \{1, 2, \ldots, \}. We assume given an increasing sequence of times

$$0 \leq t_0 < t_1 < \cdots,$$

such that during the time interval $[t_{k-1}, t_k]$ we have $P = P^k$, $I = I^k$ in System (3), (4) (and hence also in System (6), (7)).

We make the following assumptions, in terms of parameters $T > 0$, $\varepsilon > 0$, $N \geq 0$, $R > 1$, $\lambda > 0$:

**Hypothesis 4.1.**

**Unit vectors:** $\|P\| = 1$ for all $P \in \mathcal{P}$.

**Separation of patterns:** $|P \cdot Q| < \varepsilon$ for distinct $P, Q \in \mathcal{P}$.

**Duration of presentation:** $t_k - t_{k-1} \geq T$ for all $k$.

**Frequency of presentation:** Each pattern appears at least once among every $N + 1 \geq 1$ successive input patterns

$$P^j, \ldots, P^{j+N+1}, \quad j \in N_+.$$

**Initial values:**

$$|y(t_0)| < 1, \quad \|M(t_0)\| < R,$$

where $\|M\| = \sqrt{M \cdot M}$ denotes the Euclidean norm of vector $M$.

It is easy to see that the region $|y| < 1$ is invariant for System (6), (7). Henceforth we assume $|y(t)| < 1$.

**Slow adaptation:** The learning-rate parameter $\lambda > 0$ in System (3), (4) must be sufficiently small.

The following theorems refer to System (3), (4). They mean, first, that parameters can be chosen so that variables stay within given bounds, and second, that whenever the pattern from a consistent pair is input after being properly trained once, the output is always correct, whether or not a training signal is input, and regardless of the training signals when other patterns are input.

The quantity

$$\theta = \varepsilon N (1 + \varepsilon)^N$$

plays a key role.
Theorem 4.2. There exists $T_* > 0$, depending on $\lambda$, with the following properties. Assume that Hypothesis 4.1 holds with $T \geq T_*$ and

$$\theta < \frac{R - 1}{R + 1}. \quad (12)$$

Then:

$$|M(t) \cdot P^k| < R \quad \text{for all } t \geq 0, k = 1, 2, \ldots. \quad (13)$$

It is clear from (3) that if a vector $Q \in \mathbb{R}^d$ is orthogonal to all the input patterns, then $M(t) \cdot Q$ is constant. Therefore the conclusion of Theorem 4.2 implies that $|M(t)|$ is bounded.

It is worth noting that (12), or even the weaker assumption that $|P^\alpha \cdot P^\beta| < 1/N$ for $\alpha \neq \beta$, implies that $\mathcal{P}$ is a linearly independent set of vectors. This can be seen, e.g., by applying Gershgorin's circle theorem [13] to the matrix of inner products.

A pair $(P, I)$ is called eventually consistent provided:

(a) $I \in \{-1, 1\}$,

(b) there exists $n(P, I) \in \mathbb{N}_+$ such that

(i) $(P^k, I^k) = (P, I)$ for $k = n(P, I)$; and

(ii) if $k \geq n(P, I)$ and $P^k = P$, then $I^k \in \{I, 0\}$.

In other words, at some time $P$ is trained on $I$, and never subsequently trained on the "wrong" output $-I$.

Theorem 4.3. Let $\rho > 0$ be given. There exists a $T_* > 0$ with the following properties. Assume the hypothesis of Theorem 4.2., and in addition

$$N \varepsilon < \frac{1 - \delta - \kappa^{-1}}{R + 1}, \quad (14)$$

where $\kappa > 1$ is the gain in the sigmoid. Then for any eventually consistent pair $(P, I)$ we have

$$|y(t_k) - I| < \rho$$

for all $k \geq n(P, I)$ such that $P^k = P$.

Explicit estimates for $T_*$ can be derived from the proofs. Here we note only that $T_* \to \infty$ as any of the following happen: $\lambda \to 0$, $\kappa \to \infty$, $\rho \to 0$, $R \to \infty$, or $R \to 1$.

5. Discussion

It is very important for our results that $T_*$ does not depend on the choice of a particular consistent pair $(P^z, I^z)$, nor on the particular input patterns. There can be
any number (up to \( N + 1 \)) of patterns, and the same \( T_* \) will work simultaneously for all of them. Note, however, that the duration of presentation \( T_* \) explodes as the desired precision \( \rho \) goes to zero.

Any subset \( \mathcal{P}_0 \subset \mathcal{P} \) of consistently trained input patterns yields the "correct" output, regardless of the training of the other patterns, which may be untrained or even inconsistently trained. Moreover, the system can be trained or retrained on any pattern, without disturbing the previous training of the other patterns.

The requirement that input patterns be unit vectors is made only for simplicity. If they are not unit vectors then the assumption on separation of distinct patterns in Hypothesis 4.1 is changed to

\[
\frac{|P^* \cdot P^\beta|}{\|P^\beta\|} < \epsilon,
\]

with similar results.

While the assumption of near orthogonality is severe, it is interesting to note that if a fixed number of unit vectors \( P^\alpha \) are chosen independently at random in \( \mathbb{R}^d \), then as \( d \to \infty \) the expected value of \( \max_{\alpha \neq \beta} |P^\alpha \cdot P^\beta| \) tends to 0. There are also ways of preprocessing a set of linearly independent vectors to orthogonalize them (e.g., [15]). Numerical simulations with small numbers of patterns suggest that learning will take place even if the hypothesis of near orthogonality is greatly relaxed, provided the presentation time \( T \) is sufficiently enlarged.

The dimension \( d \) of the input space is irrelevant to our results, except that the number of patterns that can be learned by this method is bounded by \( d \), by the requirement of near orthogonality. In fact the input patterns could be members of an arbitrary inner product space.

Practically any sigmoid can be used in place of the ramp function \( g \), with similar results; in fact many other dynamical systems can be treated analogously.

Training signals \( I \) that are "postsynaptic" instead of "presynaptic" can be used. That is, (1) can be changed to:

\[
\frac{dy}{dt} = -y + g(M \cdot P) + I,
\]

with the corresponding changes in the other equations.

It is not hard to see that some restrictions on the geometry of the patterns is necessary. For example, a set \( \mathcal{P} \) of linearly dependent patterns strongly constrains the associated outputs that can be learned. It is also easy to give examples showing that if a pattern is not repeated within bounded times, it may be forgotten.

Many interesting stochastic questions arise. Suppose, for example, that instead of consistent pairs, we consider more generally input–output pairs \((P^\alpha, I^\alpha)\) for which there is some small probability that the wrong teaching signal \(-I^\alpha\) is given while \( P = P^\alpha \). It is true, as seems likely, that under some similar training scheme there is a high probability of correct output? This has been answered affirmatively in the adaptive filtering framework [4, 12, 13].
6. Proofs of theorems

To prove Theorem 4.2 it suffices to consider \( t \in [t_0, t_{N+1}] \), \( 1 \leq k \leq N + 1 \), for then the same argument applies to arbitrary ranges of the form \([t_{i-1}, t_{i+N}]\), \( i \leq k \leq i + N \). Similarly, for Theorem 4.3.

By (12) we choose \( \delta > 0 \) such that
\[
1 + \delta + (R + 1)\delta < R
\]
and
\[
\delta < (1 + \varepsilon)R - 1.
\]
Assume henceforth \( T_\ast \) is so large (depending on \( \lambda \)) that if in System (6), (7) we have
\[
|x(0)| < R, \quad |y(0)| < 1, \quad T \geq T_\ast,
\]
then
\[
|x(T)| < 1 + \delta.
\]
Then if \( P_k = P^x \) we have
\[
|x^x(t_k)| < (1 + \varepsilon)(R + 1) - 1,
\]
by (17) and (16). Set
\[
M_0 = \max\{1, \max\{|x^x(t_0)| : P^x \in \mathcal{P}\}\}.
\]
For \( k \in \mathbb{N}_+ \) define
\[
M_k = \max\{1, \max\{|x^x(t)| : P^x \in \mathcal{P}, t \in [t_{k-1}, t_k]\}\}.
\]
Let \( k \in \mathbb{N}_+ \), with \( P_k = P^\beta \). Then the dynamics of System (6), (7) shows that \( x^\beta(t) \) moves from \( x^\beta(t_{k-1}) \) toward \( I_k \in \{-1, 0, 1\} \) as \( t \to t_k \) in \([t_{k-1}, t_k]\). Therefore
\[
|x^\beta(t) - x^\beta(t_{k-1})| < |x^\beta(t_{k-1})| + 1,
\]
whence
\[
|x^\beta(t) - x^\beta(t_{k-1})| < M_{k-1} + 1.
\]
From (11) (with \( \beta, \alpha \) interchanged) we therefore find: for all \( P^x \in \mathcal{P} \) such that \( P^x \neq P^k \) we have, for all \( t \in [t_{k-1}, t_k] \):
\[
|x^x(t) - x^x(t_{k-1})| < \varepsilon(M_{k-1} + 1),
\]
and therefore
\[
|x^x(t)| < |x^x(t_{k-1})| + \varepsilon(M_{k-1} + 1).
\]
Therefore if \( M_k \neq |x^\beta(t)| \), then we have
\[
M_k < M_{k-1} + \varepsilon(M_{k-1} + 1),
\]
or equivalently
\[
M_k + 1 < (1 + \varepsilon)(M_{k-1} + 1). \tag{20}
\]

We now prove by induction:
\[
M_k + 1 < (1 + \varepsilon)^k(R + 1). \tag{21}
\]

For let \( P_k = P^\beta \), \( M_k = |x^\beta(t_k)| \). If \( \beta \neq \alpha \) then (20) completes the induction. If \( \beta = \alpha \) we simply quote (18).

Now fix a pattern \( P^\alpha \in \mathcal{P} \). There is a maximal \( l \in \{1, \ldots, N\} \) for which \( P^\alpha = P_l \).

By (17)
\[
|x(t)| < 1 + \delta, \quad (t \in [t_l-1, t_l]).
\]
Since \( P_k \neq P^\alpha \) for \( l + 1 \leq k \leq N + 1 \), by repeated applications of (19) and (21) we find that for \( t \in [t_l, t_{N+1}] \) we have:
\[
|x^\alpha(t)| - (1 + \delta) < \varepsilon \sum_{j=1}^{N} (1 + \varepsilon)(M_j + 1) < \varepsilon \sum_{j=1}^{N} (1 + \varepsilon)^j(R + 1) = (R + 1)(1 + \varepsilon)^j[(1 + \varepsilon)^{N-l+1} - 1].
\]

Estimating the term in brackets by the mean value theorem we find
\[
|x^\alpha(t)| - 1 - \delta < (R + 1)\varepsilon(N - l + 1)(1 + \varepsilon)^N,
\]
and therefore
\[
|x^\alpha(t)| < 1 + \delta + (R + 1)\varepsilon N(1 + \varepsilon)^N = 1 + \delta + (R + 1)\theta < R
\]
by (15). This proves Theorem 4.2.

To prove Theorem 4.3, we assume \( \delta \) and \( T_\varepsilon \) satisfy the preceding conditions, implying in particular that \( |x(t)| < R \) in System (6), (7). By (14) we may also assume \( \delta \) small enough so that
\[
N\varepsilon(R + 1) < 1 - \kappa^{-1} - \delta.
\]
Suppose in System (6), (7) that \( P = P_k = P^* \) and \( I = I_k \in \{-1, 1\} \). Assume \( |x^a(t_{k-1})| < R \). Then we choose \( T_* \) large enough (depending on \( \lambda \)) that \((x^a(t_k), y(t_k))\) is so close to the unique equilibrium at \((I, I)\) that

\[
|x^a(t_k) - I_k| < \delta < 1 - \kappa^{-1} \quad \text{and} \quad |y(t_k) - I_k| < \rho.
\]

Assume now that a pair \( (P_a, I_a) \) is eventually consistent. Let \( k \geq n(P, I) \) with \( P_k = P^* \). We must ensure, at the next \( j > k \) such that \( P_j = P^* \), that \((x^a(t_{j-1}), y(t_{j-1}))\) is in the basin \( \mathcal{B}(I_a, I_a) \) for the dynamics of System (6), (7) with \( P = P^*, I = I_j \), even when \( I_j = 0 \) or \(-I_a\). The case \( j = k + 1 \) is trivial, so we assume \( j \geq k + 2 \).

We will use the following estimate, whose proof is omitted: Provided \( \lambda < 1 \),

\[
(x, y) \in \mathcal{B}(I_a, I_a) \quad \text{if} \quad |I_a - x| < 1 - \kappa^{-1}, \quad |y| \leq 1.
\]

Let

\[
j = k + r, \quad 2 \leq r \leq N,
\]

with

\[
P_j = P^*, \quad P_i \neq P^* \quad \text{for} \ k < i < j.
\]

It follows from Theorem 3.2 (with indices interchanged) and (14) that

\[
|x^a(t_{j-1}) - x^a(t_k)| < r \varepsilon (R + 1) < N \varepsilon (R + 1).
\]

Therefore

\[
|x^a(t_{j-1}) - I_a| < |x_a(t_k) - I_a| + (N - 1) \varepsilon (R + 1) < \delta + (N - 1) \varepsilon (R + 1) < 1 - \kappa^{-1}.
\]

The proof is completed by (22).

References


Morris W. Hirsch received his doctorate at the University of Chicago in 1958. He is a Professor of Mathematics at Berkeley, where he has worked since 1960. His books include Differential Topology, and Differential Equations, Dynamical Systems, and Linear Algebra (with S. Smale). He is the author of a chapter on dynamics in Mathematical Perspectives on Neural Networks (P. Smolensky, M. Mozer and D. Rumelhart (Eds.), Lawrence Erlbaum, Mahwah, NJ, 1996).