ON THE USE OF NEURAL NETWORKS IN THE ANALYSIS OF NONLINEAR SYSTEMS: REALIZATION, APPROXIMATION, AND FEEDBACK CONTROL

Héctor J. Sussmann[1]
Department of Mathematics, Rutgers University, New Brunswick, NJ 08903
E-mail: sussmann@hamilton.rutgers.edu

1 Introduction

One of the most challenging issues in mathematical modeling is to find good classes of functions that can be used to represent general, unknown nonlinear functions. Usually, one is interested in *parametrized classes*, i.e. collections of functions depending on a certain finite number of parameters that one then tries to "fit" so as to obtain a good representation of a desired function f. The function f itself may be part of the description of the behavior of some given system —in which case, typically, one can observe the values of $f(x)$ for various "input" vectors x— or it may be part of a system to be *designed*, e.g. a feedback controller $x \to u = f(x)$. In the former case, the goal is to carry out a *nonlinear regression*. In the latter, to "tune" a system by adjusting parameters so as to produce a desired behavior.

Recently, the class of functions defined by *Neural Networks* has been proposed as a tool for parameter fitting in the two senses described above, and a number of reasons have been suggested to justify this choice. The purpose of this note is to survey some recent theoretical results that bear on the question whether this proposal is a reasonable one. Our final assessment will be cautiously optimistic: although the evidence at the moment is still inconclusive, some recent results —especially those of A. Barron— suggest that indeed neural nets may turn out to be superior in some ways to other procedures for representing nonlinear functions.

2 Neural Nets as a tool for nonlinear regression

The goal of *nonlinear regression* is to estimate unknown nonlinear functions f of any number d of variables $x_1, x_2, \ldots, x_d$, by trying to fit through a set of data for a given f some explicitly given nonlinear function $g(x_1, x_2, \ldots, x_d, \theta)$ involving a vector parameter θ. One tries to estimate θ so as to get a fit which is good according to some error criterion.

[1]This work was supported in part by the National Science Foundation under NSF Grant DMS-8902994 and by the U.S. Air Force under grant AFOSR 91-0343.

More precisely, one assumes given, for each d, a sequence of functions $\{g_{d,j}\}_{j=1}^{\infty}$, where each $g_{d,j}$ is a function $(x_1, x_2, \ldots, x_d, \theta_1, \ldots, \theta_{\nu(j)}) \rightarrow g(x_1, x_2, \ldots, x_d, \theta_1, \ldots, \theta_{\nu(j)})$ of $d + \nu(j)$ variables, the first d of which are thought of as "inputs," while the remaining $\nu(j)$ are "parameters." One can then ask, for a given function f in some class, to approximate f in some norm by the function $x \rightarrow g(x, \theta)$ obtained from g by choosing some value for the parameter θ. Alternatively, one may ask to *interpolate* a function of the form $x \rightarrow g(x, \theta)$ through a given finite set of data, i.e. to find, for a given finite set F of pairs $(x, y) \in \mathbb{R}^d \times \mathbb{R}$, a value of θ such that $y = g(x, \theta)$ for all $(x, y) \in F$.

In the neural network literature, it has been suggested that neural nets are particularly suited to this task because of the various "universal approximation theorems," according to which any function f can be approximated by functions computed by feedforward nets with a single hidden layer. (This has been shown to be true for various classes $\mathcal{F}$ of functions f and various ways of measuring the error. For example, it is true when $\mathcal{F}$ is the set of continuous functions on a compact subset K of $\mathbb{R}^d$, and the error is the sup norm, or when $\mathcal{F}$ is the space $L^2(K)$ of square-integrable functions on K, and the approximation is understood in the L^2 sense, cf. [3], [5].)

The best "universal approximation theorem" known to date is a result due to M. Leshno. Let us define a "universal approximator" to be a function $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ that has the following property:

(UA) for every d, and every continuous real-valued function f on $\mathbb{R}^d$, there is a sequence of functions $f_j : \mathbb{R}^d \rightarrow \mathbb{R}$ that converges to f uniformly on compact sets, such that each f_j is of the form $f_j(x) = a_0 + \sum_{k=1}^{N} a_k\, \sigma(\lambda_k^0 + \langle \lambda_k, x \rangle)$, where the a_k and the λ_k^0 are real numbers, and the λ_k are vectors in $\mathbb{R}^d$.

Then Leshno's Theorem says

Theorem 1 *Let $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ be a function which is bounded on every compact interval $[a, b]$, and such that the set of points of discontinuity of σ has Lebesgue measure zero. Then σ is a universal approximator if and only if σ is not a polynomial.* ∎

As justifications for the systematic use of neural networks as tools for nonlinear regression, the universal approximation theorems are of questionable significance. Indeed, similar theorems are also true of many other classes of approximating functions g, e.g. polynomials or trigonometric sums. So, if neural networks are to be shown to be superior to other methods, there should be other reasons.

One such reason was proposed by Barron in [1], [2], where he suggested that we compare the number m of parameters that we need to incorporate in a model $g(x_1, x_2, \ldots, x_d, \theta_1, \theta_2, \ldots, \theta_m)$ in order to achieve an *a priori* specified degree of accuracy. Barron showed that, for certain special classes Γ_c (which also depend on d), if $m(k, c, d)$ is the number of neural network weights that are needed in order to approximate every $f \in \Gamma_c$ in mean-square norm within error $\frac{1}{k}$ by a function g computed by a neural net G, then $m(k, c, d)$ behaves, as $k \rightarrow \infty$, like a constant times k^2, and this constant is linear in d. Barron points out that, for very similar classes such as the Sobolev spaces H^1 of square-integrable functions with square-integrable first partial derivatives, any approximation by a family of functions depending *linearly* on the parameters requires a much larger number of parameters, which grows like a high —d-dependent— power of k for fixed d, and exponentially in d as $d \rightarrow \infty$. In this way, neural network approximation can be said to require significantly fewer parameters than all approximations by families of linear subspaces, and to beat the "curse of the dimension" to which these other linear methods are subjected.

However, the significance of the above result is less than fully transparent, due to the fact that the spaces for which the other methods do badly are not exactly the same as the one for which Barron proves his positive result. For this reason, it is important to supplement Barron's theorem with another, more recent observation, also due to Barron, according to which a negative result holds for linear approximations *for the same space where Barron's positive results apply.* For simplicity, we will state the results for a space slightly different from Barron's, namely a subspace of L^2 of a cube defined by a condition given in terms of the Fourier series of a function, rather than in terms of its Fourier transform as in Barron's work. For this space, it is easy to verify that the proof of Barron's positive result still applies with practically no change, and to prove a negative result showing that, for all possible approximations by increasing families of linear subspaces, the minimum number of parameters behaves exponentially in d, and like $k^{\frac{d-1}{2}}$ for fixed d. This shows that, for the spaces considered here, neural network approximations require significantly fewer parameters than any possible linear approximation, as long as $d \geq 6$.

To state Barron's results precisely, let us use I to denote the interval $[0,1]$, and I^d to denote the d-dimensional unit cube, i.e. the product $[0,1]^d$. We will work in the space $L^2(I^d)$ of real-valued square-integrable functions on I^d.

Every $f \in L^2(I^d)$ has a Fourier series expansion $f = \sum \nu f_\nu \varphi_\nu$. Here the indices ν run over the set $M(d)$ of all d-dimensional multiindices $\nu = (\nu_1, \ldots, \nu_d)$, where the ν_j are (positive or negative) integers. The functions φ_ν are the complex exponentials, i.e. $\varphi_\nu(x) = e^{2\pi i\langle\nu,x\rangle}$. The coefficients f_ν are given by the formula $f_\nu = \int_{I^d} f(x)e^{-2\pi i\langle\nu,x\rangle}dx$. The Fourier series converges in the mean square sense, i.e. $\lim_{N\to+\infty} ||f - \sum_{||\nu||\leq N} f_\nu\varphi_\nu||_{L^2} = 0$. Here $||f||_{L^2}$ denotes the L^2 norm of f, i.e. $||f||_{L^2} = (\int_{I^d} |f(x)|^2 dx)^{\frac{1}{2}}$ and, for each multiindex $\nu \in M(d)$, we define $||\nu|| = |\nu_1| + \ldots + |\nu_d|$.

If $c > 0$, we let

$$\Gamma(c,d) = \{f \in L^2(I^d) : \sum_\nu ||\nu||\,|f_\nu| \leq c\} \,. \tag{1}$$

Then it is clear that every element of $\Gamma(c,d)$ is continuously differentiable on I^d.

Barron's results (cf. [2]), reformulated in terms of the spaces $\Gamma(c,d)$, are as follows:

Theorem 2 *Let $\sigma : \mathbb{R} \to \mathbb{R}$ be an arbitrary sigmoid function. Let $N(k,c,d)$ be the smallest integer L that satisfies the inequality $L > 16\pi^2c^2k^2$. Then for every f in $\Gamma(c,d)$ and every integer $k > 0$ there exists a single-layer feedforward neural network G with d inputs, one output, transfer function σ, and a number N of hidden nodes not greater than $N(k,c,d)$, such that the map g computed by G satisfies $||f - g||_{L^2} \leq \frac{1}{k}$.*

Theorem 3 *Let $S_1, S_2, S_3, \ldots$ be a sequence of finite-dimensional linear subspaces of $L^2(I^d)$, of dimensions $m_1, m_2, m_3, \ldots$, with the property that whenever $f \in \Gamma(c,d)$ and k is a positive integer, then there exists an element $g \in S_k$ such that $||f - g||_{L^2} \leq \frac{1}{k}$. Then the inequality $m_k \geq p_{lin}(k,c,d)$ holds for all $k = 1, 2, 3, \ldots$, where*

$$p_{lin}(k,c,d) = \frac{(L+d)!}{L!d!} \;,\; L = [ck] - 2\,. \tag{2}$$

(In the above statement we use the standard notation $[r]$ for the *integral part* of the real number r, i.e. the largest integer n such that $n \leq r$.)

We recall that a *sigmoid function* is —following the definitions of Barron's papers cited above— a continuous increasing function σ from $\mathbb{R}$ to $\mathbb{R}$ such that $\lim_{t\to+\infty} \sigma(t) = 1$ and $\lim_{t\to-\infty} \sigma(t) = 0$.

A *single-layer neural net with d inputs and one output* consists of the specification of a number N of *hidden nodes*, labeled by an index $i \in \{1, 2, \ldots, N\}$, together with a sigmoid "transfer function" σ, an array $W = \{w_{ij}\}_{1\leq i\leq N, 0\leq j\leq d}$ of *incoming weights for the hidden nodes*, and an array $U = \{u_i\}_{0\leq i\leq N}$ of *outgoing weights.* The function g computed by a net G is then the function g given by

$$g(x_1, \ldots, x_d) = u_0 + \sum_{i=1}^{N} u_i \sigma\Big(w_{i0} + \sum_{i=1}^{d} w_{ij} x_j\Big) . \tag{3}$$

So, a net with N hidden nodes involves $1 + N(d + 2)$ parameters. Theorem 2 can therefore be reinterpreted in the following way, that displays the analogy with Theorem 3. Let $\mathcal{N}_k(c, d)$ be the "nonlinear submanifold" of $L^2(I^d)$ consisting of those functions that can be computed with a neural net as above with $N(k, c, d)$ hidden nodes. Let $p_{nn}(k, c, d) = 1 + N(k, c, d)(d + 2)$, i.e.

$$p_{nn}(k, c, d) = 1 + (1 + [16\pi^2 c^2 k^2])(d + 2) . \tag{4}$$

Then the sets $\mathcal{N}_k(c, d)$ are parametrized by $p_{nn}(k, c, d)$ parameters, and have the same approximation properties as the linear subspaces S_k considered in Theorem 3. However, Theorem 3 says that, no matter how the linear subspaces S_k are selected, the approximation property cannot be satisfied unless S_k involves at least $p_{lin}(k, c, d)$ parameters. Clearly, $p_{nn}(k, c, d) << p_{lin}(k, c, d)$ in various ways. (For example, $p_{nn}(k, c, d)$ behaves as $k \to \infty$, for each fixed d, like a constant times k^2, whereas $p_{lin}(k, c, d)$ goes like a constant times k^d, which is a faster growth as long as $d \geq 3$. Also, if $d \to \infty$ and k is fixed, then $p_{nn}(k, c, d)$ grows linearly in d, but $p_{lin}(k, c, d)$ grows like d^k.) This shows that, at least for the particular type of approximation considered here (L^2 approximation of functions belonging to $\Gamma(c, d)$) neural nets do much better than *any* approximation by linear subspaces.

The question whether these results indeed establish that "neural nets beat the curse of the dimension" is not yet settled, however. This is because the spaces where the positive results apply are only described in terms of properties of the Fourier series (in the version used here) or Fourier transform (in Barron's version) of the functions, which does not lend itself to a simple interpretation. For this reason, it is not clear whether these spaces can be said to be, in some sense, "independent of the dimension." (Such independence should be an obvious requirement. It would not do much good to show that a particular method beats the curse of the dimension if this was done by selecting, for each d, a space $\mathcal{F}_d$ of functions that somehow becomes nicer and nicer as d-grows, so that its approximation properties get better and better exactly as needed so as to compensate for the increase in dimension.) Barron's results show that, at least for some classes of functions, the "curse of the dimension" cannot be beaten using linear approximations whereas. *for the same class*, neural networks do beat it.

3 Uniqueness theorems

We now turn to the following question: to what extent is a feedforward net uniquely determined by its corresponding input-output map? Hecht-Nielsen pointed out in [4] that there are some obvious sources of non-uniqueness, arising from internal symmetries such as the possibility of relabeling the hidden nodes. He asked whether the small, discrete symmetry group $\mathcal{G}$ that he exhibited was actually the full symmetry group, i.e. whether the input-output map of the net determines the net modulo the symmetries in $\mathcal{G}$.

We have given an affirmative answer to this question, by proving a uniqueness theorem: two "irreducible" nets with the same input-output map are related by a transformation in $\mathcal{G}$.

The irreducibility condition is needed because there is another source of nonuniqueness, coming from the fact that a net may contain nodes that make no contribution whatsoever to the output, e.g. nodes whose outgoing connection weights are equal to zero. This means that there are cases when a net can be *reduced* (i.e. some of its nodes can be removed) without changing the input-output map. Clearly, a uniqueness theorem can only hold for *irreducible nets.*

The uniqueness result applies to feedforward nets with a single hidden layer, and a transfer function $\sigma(s) = \operatorname{Tanh} s$. For such nets, we have been able to give a precise definition of reducibility and of the group $\mathcal{G}$ of internal symmetries, and to show rigorously that an irreducible net is uniquely determined, up to symmetries in $\mathcal{G}$, by its input-output map.

As a byproduct of these results, it follows that *an irreducible net is minimal.* (We call a net "minimal" if its input-output map cannot be obtained from another net with fewer hidden nodes. So minimality is not, in principle, an easily checkable property. On the other hand, "irreducibility" is defined by listing three very simple situations under which a net could be made smaller, and then calling the net irreducible if none of these situations occurs. So irreducibility is easily checked.) The fact that irreducibility is equivalent to minimality means that there is no mechanism, other than the three listed in the definition of reducibility, that can be used to reduce the number of hidden nodes without changing the input-output map.

To state the uniqueness result precisely, we consider feedforward nets with m input nodes, one layer of hidden neurons, one output node, and "transfer function" $\sigma(s) = \operatorname{Tanh} s$, so we will just use the word "net" to refer to this particular kind of neural net. The set of all the nets in this sense that have n hidden nodes will be denoted by $\mathcal{N}_{m,n}$, so that

1. $\mathcal{N}_{m,n}$ is the set of all networks with m input units, one output unit, n hidden units, and activation function Tanh,
2. a *net* is a member of $\mathcal{N}_{m,n}$ for some m, n.

Precisely, then, a *net* consists of the specification of (a) m *input nodes*, labeled $1, \ldots, m$, (b) n *hidden nodes*, labeled $1, \ldots, n$, (c) one *output node*, (d) an array $W = \{w_{ij}\}$ of weights for the connections from input nodes to hidden nodes, and (e) an array $C = \{c_j\}$ of weights for the connections from the hidden nodes to the output node. The w_{ij} are indexed by $i \in \{0, \ldots, m\}$, $j \in \{1, \ldots, n\}$, so that w_{ij} is the weight of the connection from the i-th input node to the j-th hidden node, and w_{0j} is the "bias" for the j-th hidden node. Similarly, the c_j are indexed by $j \in \{0, \ldots, n\}$.

Given an input vector $x = (x_1, \ldots, x_m) \in \mathbb{R}^m$, the *net input* $\nu_j(x)$ of the j-th hidden node is the number $\nu_j(x) = w_{0j} + \sum_{i=1}^m w_{ij} x_i$. The *output* $y_j(x)$ of the j-th hidden node is the number $y_j(x) = \sigma(\nu_j(x)) = \operatorname{Tanh}(\nu_j(x))$. The *net input* of the output node is $\mu(x) = c_0 + \sum_{j=1}^m c_j y_j(x)$. The *output* $z(x)$ of the net is the number $z(x) = \sigma(\mu(x))$. The function $x \to z(x)$ is the *input-output map* of the net.

For a fixed m, two nets are *I-O equivalent* if their corresponding input-output maps are the same. Our uniqueness result, proved in [17], states that two I-O equivalent nets must necessarily be the same net, up to some simple internal symmetries.

Fix m. To a net with n hidden nodes we associate the n linear affine functions $\mathbb{R}^m \ni x \to \nu_j(x) \in \mathbb{R}$, $j = 1, \ldots, n$. We call two linear affine functions α, β on $\mathbb{R}^m$ *sign-equivalent* if $|\alpha(x)| = |\beta(x)|$ for all $x \in \mathbb{R}^m$. (This is easily seen to be equivalent to the condition that either $\alpha(x) = \beta(x)$ for all x or $\alpha(x) = -\beta(x)$ for all x.) We call a net *reducible* if one of the following conditions hold:

(I) one of the c_j, for $j = 1, \ldots, n$, vanishes;

(II) there exist two different indices $j_1, j_2 \in \{1, \ldots, n\}$ such that the functionals ν_{j_1}, ν_{j_2} are sign-equivalent.

(III) one of the functionals ν_j is a constant.

If a net N is reducible, then it is I-O equivalent to another net with fewer hidden nodes. Indeed, if N is reducible because (I) holds, and $j \in \{1, \ldots, n\}$ is such that $c_j = 0$, then the j-th node makes no contribution to the net input $\mu(x)$, so we can simply remove it without changing the output. Now suppose that N is reducible because (II) holds. Let j_1, j_2 be such that ν_{j_1} and ν_{j_2} are sign-equivalent. Let $\nu_{j_1}(x) \equiv \rho\nu_{j_2}(x)$, where $\rho = 1$ or $\rho = -1$. Then the combined contribution of the nodes j_1 and j_2 to $\mu(x)$ is $c_{j_1}\sigma(\nu_{j_1}(x)) + c_{j_2}\sigma(\nu_{j_2}(x))$, i.e. $c_{j_1}\sigma(\rho\nu_{j_2}(x)) + c_{j_2}\sigma(\nu_{j_2}(x))$, which is equal to $\rho c_{j_1}\sigma(\nu_{j_2}(x)) + c_{j_2}\sigma(\nu_{j_2}(x))$, i.e. to $(\rho c_{j_1} + c_{j_2})\sigma(\nu_{j_2}(x))$. (Here we used the fact that σ is an odd function, so $\sigma(\rho s) = \rho\sigma(s)$.) So we could obtain the same input-output map by just removing Node j_1 and changing the weight of the connection from Node j_2 to the output from c_{j_2} to $c'_{j_2} = \rho c_{j_1} + c_{j_2}$. Finally, if N is reducible because (III) holds, and ν_j is constant with value κ, then we can remove the j-th hidden node and change c_0 to $c_0 + \sigma(\kappa)$.

An net which is not reducible will be called *irreducible*. A net with n hidden nodes will be called *minimal* if it is not I-O equivalent to a net with fewer hidden nodes. We have shown above that a minimal net is necessarily irreducible. It will follow from our uniqueness theorem that, conversely, an irreducible net is minimal.

The are some obvious transformations that can be applied to a net N without changing its input-output map. For instance, suppose we pick a hidden node j and change the sign of all the weights w_{ij} for $i = 0, 1, \ldots, m$, and also change the sign of c_j. Since σ is odd, this will not alter the contribution of this node to the total net input $\mu(x)$. Let us use $\theta_j(N)$ to denote the net resulting from this transformation.

Another possibility is to interchange two hidden nodes, that is, to take two hidden nodes j_1 and j_2 and relabel j_1 as j_2 and j_2 as j_1, taking care to also relabel the corresponding weights. (Precisely: consider a new net with weights w_{ij}^{new}, c_j^{new}, such that $w_{ij_1}^{new} = w_{ij_2}$, $w_{ij_2}^{new} = w_{ij_1}$, $c_{j_1}^{new} = c_{j_2}$, $c_{j_2}^{new} = c_{j_1}$.) Call the resulting net $\tau_{j_1j_2}(N)$. The maps $\tau_{j_1j_2}$, θ_j generate a finite group $\mathcal{G}_{m,n}$ of transformations of the set $\mathcal{N}_{m,n}$, with $2^n n!$ elements. Call two nets in $\mathcal{N}_{m,n}$ *equivalent* if they are related by a transformation in $\mathcal{G}_{m,n}$. Then two equivalent nets are clearly I-O equivalent. The uniqueness theorem (cf. [17]) says that the converse is true for irreducible nets:

Theorem 4 *Let N_1, N_2 be irreducible I-O equivalent nets in $\mathcal{N}_{m,n_1}$, $\mathcal{N}_{m,n_2}$. Then (i) $n_1 = n_2$ and (ii) N_1 and N_2 are equivalent.* ∎

An important corollary of Theorem 4 is the following observation:

An irreducible net is minimal.

We remark that the above results are very limited in two ways, namely, (a) in that they only apply to the very special transfer function $\sigma(s) = \operatorname{Tanh} s$, and (b) in that they have only been established for single-layer nets. It is an important open problem to extend the uniqueness theorems to general transfer functions and feedworward nets with a more general graph of interconnections.

4 Neural nets as tools for systems stabilization

An important issue in control is to find, for a control system given by a differential equation $\dot{x} = f(x,u)$ (where $x \in \mathbb{R}^d$ is the "state" of the system, and $u \in \mathbb{R}^m$ is the "control," a control function $x \to u(x)$ (known as a "feedback control," or "closed-loop control,") such that that the "closed-loop system" $\dot{x} = f(x, u(x))$ has some desired behavior. One of the most commonly desired behaviors is, simply, that a certain point $\bar{x}$ —which we may, for simplicity, always choose to be the origin of $\mathbb{R}^d$ be an *asymptotically stable equilibrium* (at least locally, and if possible also globally). A control law for which these properties hold will called a (locally, or globally) *stabilizing feedback.*

In order to "find" a stabilizing feedback, one often uses algorithms where it is posited *a priori* that the feedback law will be given in the form $u = g(x, \bar{\theta})$, where g is a given function of x and the parameter θ, and $\bar{\theta}$ is a value of θ to be determined. Once again, the problem of finding an arbitrary function is replaced by that of finding a value of the parameter.

Neural networks have turned out to be useful in two ways. First, for the class of *linear systems*, for which there is a well established theory of stabilization by linear feedbacks, it is often desirable to use *saturated elements* to implement the feedback law. In our work [16], we gave a complete characterization of those linear systems that can be stabilized by means of a bounded feedback, and showed that for such systems there always exists a stablizing feedback in the form of a neural network of a particularly simple form. Precisely, consider a linear system $\dot{x} = Ax + Bu$, where $x \in \mathbb{R}^d$, $u \in \mathbb{R}^m$, and A, B are, respectively, a $d \times d$ and a $d \times m$ matrix. It is well known, and easy to prove, that a necessary and sufficient condition for this system to admit a globally bounded stabilizing law is that the matrix A have no eigenvalues with a strictly positive real part, and that those eigenvalues whose real part is zero be controllable. (The meaning of this last condition is as follows: if $C(A, B)$ denotes the smallest linear subspace of $\mathbb{R}^d$ that contains all the vectors Bu, $u \in \mathbb{R}^m$, and is invariant under the map $x \to Ax$, then the spectrum of the operator induced by A on the quotient $\mathbb{R}^d/C(A, B)$ cannot contain any eigenvalues with zero real part.) The main result of [16] is then the following:

Theorem 5 *Consider a system $\Sigma: \dot{x} = Ax + Bu$ such that A has no eigenvalues with positive real part, and all the eigenvalues with zero real part are controllable. Let $\sigma : \mathbb{R} \to \mathbb{R}$ be an arbitrary Lipschitz function such that $\sigma(0) = 0$, $s\sigma(s) > 0$ for $s \neq 0$, and the limits $\lim_{s\to\pm\infty} \sigma(s)$ exist and are nonzero. Then the system Σ can be stabilized by means of a feedback law given by a feedforward neural network with transfer function σ.* ∎

In [16] we actually describe the structure of such networks in great detail.

A second use of neural networks in stabilization is for the feedback stabilization of *nonlinear systems.* The nonlinear systems literature contains many results on the existence of smooth stabilizing feedback laws of various kinds. However, these results lack the remarkable simplicity of the linear theory, where, for instance, if a stablizing feedback exists at all, then it follows that there is one of a particularly simple kind, namely, a linear one. For nonlinear systems it may happen, for instance, that a system can be stabilized a by a discontinuous feedback but not by a continuous one. (For example, the system $\dot{x}_1 = u_1$, $\dot{x}_2 = u_2$, $\dot{x}_3 = u_1 x_2$ has this strange property, as was first pointed out by R. Brockett.) Moreover, the existence results do not usually provide an explicit representation of the solution, e.g. in terms of a given function with some parameters to be determined. For this reason, the following recent result of E. Sontag is particularly useful:

Theorem 6 *Let $f : \mathbb{R}^d \times \mathbb{R}^m$ be of class C^2. Assume that there exists a function $k : \mathbb{R}^d \to \mathbb{R}^m$ of class C^2 such that the origin $0 \in \mathbb{R}^d$ is an exponentially stable equilibrium of the closed-loop system $\dot{x} = f(x, k(x))$. Let K be a compact subset of the corresponding domain of attraction. Let $\sigma : \mathbb{R} \to \mathbb{R}$ be a sigmoid function of class C^2. Then there exists a feedforward neural network ν with a single hidden layer that gives rise to an input-output map $k_\nu : \mathbb{R}^d \to \mathbb{R}^m$ which has the same properties as k, that is, k_ν is of class C^2, and 0 is an exponentially stable equilibrium of $\dot{x} = f(x, k_\nu(x))$, with a basin of attraction containing K.* ▮

Theorem 6 cannot be extended to the more general class of systems that are *asymptotically controllable to the origin*, i.e. such that for every point x there exists an open-loop controller that steers x to 0 in finite or infinite time. This is so because, for such systems, continuous stabilizing laws need not exist, as shown by the Brockett example described above. Naturally, when a continuous feedback law does not exist, it will not be possible to stabilize by means of a feedforward net of any kind, with one or any number of hidden layers, as long as the net involves a continuous transfer function and therefore gives rise to a continuous input-output map. One may think, however, that if we allow *discontinuous* transfer functions, e.g. a Heaviside function, then the above result may carry through. Remarkably, this is indeed so if one allows networks with *two* hidden layers, but not if one only allows one hidden layer. Indeed, Sontag proved:

Theorem 7 *Every asymptotically controllable system can be stablized by means of a feedforward net with two hidden layers and a Heaviside transfer function. There exist asymptotically controllable systems that cannot be stabilized by means of any single-layer network, even with discontinuous transfer functions.* ▮

References

[1] Barron, A.R., "Approximation bounds for superpositions of a sigmoidal function." *Proceedings of the IEEE International Symposium on Information Theory.* IEEE Press, 1991.

[2] Barron, A.R., "Universal approximation bounds for superpositions of a sigmoidal function." Technical Report #58, Statistics Department, University of Illinois at Urbana-Champaign, 1991.

[3] Cybenko, G., "Approximation by superpositions of a sigmoidal function." *Mathematics of Control, Signals and Systems* **2** (1989), pp. 303-314.

[4] Hecht-Nielsen, R., "Theory of the Backpropagation Neural Network." *Proceedings of the International Joint Conference on Neural Networks, Washington, 1989.* IEEE Publications, NY, 593–605.

[5] Hornik, K., Stinchcombe, M., & White, H., "Multi-layer feedforward networks are universal approximators." *Neural Networks* **2** (1989), 359-366.

[6] Sontag, E.D., "Capabilities of four- vs three-layer nets, and control applications." *Proceedings of the Conference on Information Sciences and Systems.* John Hopkins University, 1991, 558-563.

[7] Sontag, E.D., "Remarks on interpolation and recognition using neural nets." *Advances in Neural Information Processing Systems 3* (R.P. Lippmann, J. Moody, and D.S. Touretzky, eds), Morgan Kaufmann, San Mateo, CA, 1991, 939-945.

[8] Sontag, E.D., "Feedback stabilization using two hidden-layer nets," in *Proc. Amer. Automatic Control Conference*, Boston, June 1991, pp. 815-820. To appear in *IEEE Trans. Neural Networks.*

[9] Sontag, E.D., "Neural nets as systems models and controllers," in *Proc. Seventh Yale Workshop on Adaptive and Learning Systems,* pp. 73-79, May 1992.

[10] Sontag, E.D., and Sussmann, H.J., "Remarks on local minima in backpropagation," *Proc. Conf. Info. Sciences and Systems,* Johns Hopkins University Press, 1989, pp. 432-435.

[11] Sontag, E.D., and Sussmann, H.J., "Backpropagation Separates when Perceptrons Do," *Neural Networks* **4**(1991), pp. 243-249.

[12] Sontag, E.D., and Sussmann, H.J., "Backpropagation can give rise to spurious local minima even for networks without hidden layers," *Complex Systems* **3**(1989), pp. 91-106.

[13] Sussmann, H.J., "On the number of memories that can be perfectly stored by a neural net with Hebb weights," *I.E.E.E. Transactions on Information Theory* **35**, No. 1 (1989), pp 786-791.

[14] Sussmann, H.J., "Learning algorithms for Boltzmann machines," *Proceedings of the 27th I.E.E.E. Conference on Decision and Control*, Austin, Texas, Dec. 1988, pp. 786-791.

[15] Sussmann, H.J., "On the convergence of learning algorithms for Boltzmann machines," Rutgers Center for Systems and Control Technical Report 88-03, August 1988. Revised version to appear in *Neural Networks.*

[16] Sussmann, H.J., Sontag, E.D., and Yang, Y., "Stabilization of linear systems with bounded controls," to appear in the *Proceedings of the 1992 IFAC NOLCOS (Nonlinear Control Symposium)* to be held in Bordeaux, France, June 1992.

[17] Sussmann, H.J., "Uniqueness of the Weights for Minimal Feedforward Nets with a Given Input-Output Map," Rutgers University Sycon Report 91-06b, May 1991, to appear in *Neural Networks.*