Wavelet Networks

Qinghua Zhang and Albert Benveniste, *Fellow, IEEE*

Abstract— **Based on the wavelet transform theory, the new notion of wavelet network is proposed as an alternative to feedforward neural networks for approximating arbitrary nonlinear functions. An algorithm of backpropagation type is proposed for wavelet network training and experimental results are reported.**

I. Introduction

THE approximation of general continuous functions by nonlinear networks such as discussed in [1], [2] is very useful for system modeling and identification. Such approximation methods can be used, for example, in black-box identification of nonlinear systems. Recently *neural networks* have been established as a general approximation tool for fitting nonlinear models from input/output data. The work of G. Cybenko [3], and Caroll and Dickinson [5] established a universal approximation property for such networks (see also [6], [7]). On the other hand, the recently introduced *wavelet decomposition* [8]–[13] emerges as a new powerful tool for approximation. Such an approximation turns out to have a structure very similar to the one achieved by a $(1+\frac{1}{2})$—layer neural network. In particular, recent advances have shown the existence of orthonormal wavelet bases, from which follows the availability of rates of convergence for approximation by wavelet based networks. This paper presents a new type of network, called a *wavelet network*, inspired by both the feedforward neural networks and wavelet decompositions. An algorithm of backpropagation type is proposed and experimental results are reported.

The paper is organized as follows. In Section I, network structures for approximation are discussed, and the wavelet network is introduced. A learning algorithm for wavelet network training is presented in Section II. Finally, experimental results are reported in Section III and conclusions are drawn.

The following generic notations will be used throughout this paper: lower case symbols such as $x, y, \alpha, \psi, \ldots$ refer to scalar valued objects, lower case boldface symbols such as $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta}, \ldots$ refer to vector valued objects, and finally capital symbols will be used for matrices.

II. Network Structures For Approximation

In this section some network structures for approximation are discussed. First, feedforward neural networks as analyzed in [3] are briefly reviewed. Then an alternative approach based on wavelet theory is presented.

Manuscript received March 19, 1991; revised March 3, 1992.

Q. Zhang is with Linköpings Tekniska Högskola, Institutionen för Systemteknik, 581 83 Linköping, Sweden

A. Benveniste is with IRISA-INRIA, Campus de Beaulieu, 35 042 Rennes Cedex, France.

IEEE Log Number 9200525.

A. Neural Networks

Fig. 1 depicts a so-called $(1+\frac{1}{2})$—layer neural network. Recently, the ability of such neural networks to approximate continuous functions has been widely studied [3], [5]–[7]. In particular, the following result has been proved in [3]:

If $\sigma(\cdot)$ is a continuous discriminatory function[1], then finite sums of the form

$$g(\boldsymbol{x}) = \sum_{i=1}^{N} w_i \sigma(\boldsymbol{a}_i^T \boldsymbol{x} + b_i) \tag{1}$$

are dense in the space of continuous functions defined on $[0,1]^n$, where $w_i,\ b_i \in \Re,\ \boldsymbol{a}_i \in \Re^n$. In other words, given any continuous function f defined on $[0,1]^n$ and any $\varepsilon > 0$, there is a sum $g(\boldsymbol{x})$ of the form (1), for which $|g(\boldsymbol{x}) - f(\boldsymbol{x})| < \varepsilon$ for all $\boldsymbol{x} \in [0,1]^n$.

Any bounded and measurable sigmoidal function[2] is discriminatory, as shown in [3] by G. Cybenko. In particular, any continuous sigmoidal function is discriminatory. Worth to be mentioned is the work [4] by Barron where tight approximation bounds for neural networks are obtained. Finally, the work of Carrol and Dickinson [5] establishes a link between best approximating neural networks and some discrete approximation of the Radon transform of the function in consideration. By the way, explicit bounds of the minimum approximation error achieved by an optimal neural network of a given size are obtained. The coefficients of the best approximating neural network are not explicitly provided by the Radon transform, however. This will be in contrast to our approach, and this has several important consequences we shall discuss later.

In proposing our wavelet networks as an alternative, we have the following objectives in mind:

- **Preserve the "universal approximation" property,** i.e., provide a class of networks exhibiting the same density property as the class (1).
- **Have an explicit link between the network coefficients and some appropriate transform.** This will be extremely useful for guessing good initial values for the backpropagation—like algorithm we shall propose. Furthermore, this may be also used for a deeper theoretical investigation of the properties of this learning algorithm, but this latter point is deferred to future work.

[1] As defined in [3], $\sigma(\cdot)$ is discriminatory if for a Borel measure μ on $[0,1]^n$, $\int_{[0,1]^n} \sigma(\boldsymbol{a}^T \boldsymbol{x} + b) d\mu(\boldsymbol{x}) = 0 \ \ \forall \boldsymbol{a} \in \Re^n$ and $\forall b \in \Re$ implies that $\mu = 0$.

[2] Defined also in [3], $\sigma(\cdot)$ is sigmoidal if

$$\sigma(t) \to \begin{cases} 1 & \text{as } t \to +\infty, \\ 0 & \text{as } t \to -\infty. \end{cases}$$

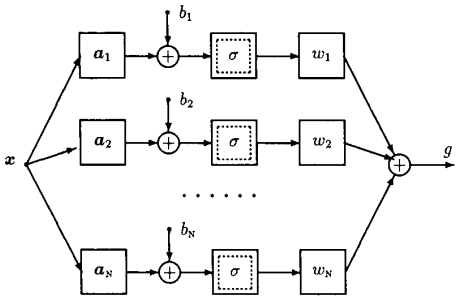


Fig. 1. $(1+\frac{1}{2})$—layer neural network for approximation. The first boxes indicate scalar products with the vectors $\boldsymbol{a}_i$. Then scalars b_i are added and sigmoid functions are applied. Finally linear combination of the outputs is taken with the w_i's as weights.

- **Possibly achieve the same quality of approximation with a network of reduced size.**

We shall discuss in our conclusion how far these objectives have been achieved. In the next subsection, the appropriate transform is introduced: the *wavelet decomposition.*

B. Wavelet Decompositions as Universal Approximants

We are interested in finding some appropriate function $\psi : \Re^n \mapsto \Re$ with the following property: there exists a denumerable family of the form[3]

$$\Phi = \Big\{ \det(D_k^{\frac{1}{2}})\psi[D_k\boldsymbol{x} - \boldsymbol{t}_k] : \boldsymbol{t}_k \in \Re^n, \quad D_k = \operatorname{diag}(\boldsymbol{d}_k), \ \boldsymbol{d}_k \in \Re_+^n, \ k \in \mathcal{Z} \Big\} \tag{2}$$

(where the $\boldsymbol{t}_k$'s are translation vectors and the $\boldsymbol{d}_k$'s are dilation vectors specifying the diagonal[4] dilation matrices D_k) satisfying the *frame* property: there exist two constants $c_{\min} > 0$ and $c_{\max} < \infty$ such that, for all f in $\mathcal{L}^2(\Re^n)$, the following inequalities hold:

$$c_{\min}\|f\|^2 \leq \sum_{\varphi \in \Phi} | < \varphi, f > |^2 \leq c_{\max}\|f\|^2 \tag{3}$$

In this sum, $< \cdot, \cdot >$ denotes the inner product in $\mathcal{L}^2(\Re^n)$ and the sum ranges over all elements of the family Φ.

It is a consequence of (3) that *the family* Φ *is dense in* $\mathcal{L}^2(\Re^n)$, and the quality of the best approximation for a fixed number of terms in the sum depends on how far from 1 the coefficients $c_{\min}$ and $c_{\max}$ are. Hence the collection of all linear combinations of elements of the frame Φ

$$g(\boldsymbol{x}) = \sum_{k=1}^{N} w_k \varphi_k(\boldsymbol{x}), \varphi_k \in \Phi, \tag{4}$$

is dense in $\mathcal{L}^2(\Re^n)$.

From this follows also that *the collection of all finite sums of the form*

$$g(\boldsymbol{x}) = \sum_{i=1}^{N} w_i \det(D_i^{\frac{1}{2}})\psi[D_i\boldsymbol{x} - \boldsymbol{t}_i] \tag{5}$$

where the $\boldsymbol{t}_i$*'s are arbitrary translation vectors and the* $\boldsymbol{d}_i$*'s are arbitrary dilation vectors specifying the diagonal dilation matrices* D_i *is also dense in* $\mathcal{L}^2(\Re^n)$, since it contains in particular all finite linear combinations of elements of the frame Φ. There are obviously more degrees of freedom in class (5) than in class (4) since translations and dilations are not constrained to belong to the denumerable families specified in (2). This additional flexibility can be used to fit those dilations and translations to a particular function, and we shall discuss this point later on.

It remains to exhibit families Φ as above: the wavelet theory will be used for this purpose.

1) Using the Continuous Wavelet Decomposition: The continuous wavelet decomposition allows us to decompose any function $f(\boldsymbol{x}) \in \mathcal{L}^2(\Re^n)$ using a family of functions obtained by dilating and translating a single *"wavelet"* function $\psi : \Re^n \mapsto \Re$. To build such a wavelet we proceed as follows. Consider first a *scalar* wavelet in the Morlet–Grossmann sense [13], [9], i.e., a function $\psi_s : \Re \mapsto \Re$ with a Fourier transform $\widehat{\psi}_s(\omega)$ satisfying the condition

$$C_{\psi_s} = \int_0^{+\infty} \frac{|\widehat{\psi}_s(\omega)|^2}{\omega} d\omega < \infty. \tag{6}$$

Using this scalar wavelet ψ_s we build the desired wavelet $\psi : \Re^n \mapsto \Re$ by setting

$$\psi(\boldsymbol{x}) = \psi_s(x_1)\ldots\psi_s(x_n) \text{for } \boldsymbol{x} = (x_1, \ldots, x_n) \tag{7}$$

and we also introduce

$$C_\psi = C_{\psi_s}^n.$$

Then the following *continuous wavelet decomposition formulae* hold:[5]

$$f(\boldsymbol{x}) = \frac{1}{C_\psi} \int_{\Re^n} \int_{\Re_+^n} \mathcal{W}(\boldsymbol{d}, \boldsymbol{t})(\det D)^{\frac{1}{2}} \psi[D(\boldsymbol{x} - \boldsymbol{t})] d\boldsymbol{d}d\boldsymbol{t} \tag{8}$$

$$\mathcal{W}(\boldsymbol{d}, \boldsymbol{t}) = \int_{\Re^n} f(\boldsymbol{x})(\det D)^{\frac{1}{2}} \psi[D(\boldsymbol{x} - \boldsymbol{t})] d\boldsymbol{x}. \quad \text{where } D = \operatorname{diag}(\boldsymbol{d}). \tag{9}$$

In these formulas, $\boldsymbol{d}$ and $\boldsymbol{t}$ are the dilation and translation vectors, respectively. This result is classical [9] for $n = 1$ and easily extends to the multidimensional case. For the sake of completeness, we give a proof for the case $n > 1$ in Appendix A. It is also shown in [9] that the denumerable family

$$\Psi_s(\alpha, \beta) \triangleq \Big\{ \alpha^{\frac{k}{2}} \psi_s(\alpha^k x - \beta l) : k, l \in \mathcal{Z} \Big\} \tag{10}$$

constitutes a frame of $\mathcal{L}^2(\Re)$ (i.e., satisfies (3)) for suitable choices of the parameters (α, β). Commonly used wavelets, such as the "Morlet" wavelet for instance, yield $c_{\min}$ and $c_{\max}$ very close to 1 for $\alpha = 2, \beta = 1$. We show in Appendix A

[3] We denote by $\Re_+^n$ the product $\Re_+ \times \ldots \times \Re_+$.

[4] More generally, the notation diag($\boldsymbol{d}$), where $\boldsymbol{d}$ is a vector, denotes the diagonal matrix built using the elements of $\boldsymbol{d}$

[5] If $\boldsymbol{x}$ is a vector, we shall use throughout this paper the notation "$d\boldsymbol{x}$" to refer to the Lebesgue measure $dx_1 \times \ldots \times dx_n$ (so $d\boldsymbol{x}$ is *not* a vector valued measure).

that we also get a frame of $\mathcal{L}^2(\Re^n)$ by taking direct products of elements of $\Psi_s(\alpha, \beta)$, i.e.,

$$\Psi(\alpha,\beta)\stackrel{\Delta}{=}\left\{(\det D)^{\frac{1}{2}}\psi(D\boldsymbol{x}-\beta\boldsymbol{l}) : \boldsymbol{d},\boldsymbol{l}\in\mathcal{Z}^n\right\}$$
$$\text{where } \boldsymbol{d}=(d_1,\ldots,d_n), D=\operatorname{diag}\left(\alpha^{d_1},\ldots,\alpha^{d_n}\right) \quad (11)$$

which is of the desired form (2).

Note that, if we take instead $\psi : \Re^n \mapsto \Re$ such that

$$C_\psi = \int_0^{+\infty} \frac{|\widehat{\psi}(h\boldsymbol{\omega})|^2}{h} dh \quad (12)$$

exists and is independent of $\boldsymbol{\omega}$ (an isotropy condition), then the wavelet decomposition formulae and frame property hold with D ranging over *scalar* matrices of the form dI [10], [11]. So isotropic wavelets can be used as well, but we shall not use this additional flexibility.

Finally, there is even no need for our purpose that the *synthesis* and *decomposition* formulae ((8) and (9), respectively) use the same wavelet ψ, different wavelets in duality can be used as well. To summarize, the continuous wavelet transform theory in the Morlet–Grossmannn sense provides us with considerable flexibility in designing our networks. In our experiments, we have selected $\psi_s = -xe^{-\frac{1}{2}x^2}$ as a scalar wavelet (it satisfies condition (6)) and in the multidimensional case we have taken direct products of such scalar wavelets. Note that the Laplacian of the Gaussian function $e^{-\frac{1}{2}|x|^2}$ satisfies the isotropic admissibility condition (12), so we might have used it directly.

2) Using Orthonormal Wavelets: On the other hand, families $\Psi(\alpha, \beta)$ satisfying the frame property (3) can be obtained directly from the theory of *orthonormal wavelets* [8], [12] by taking $\alpha = 2$ and $\beta = 1$ in (11) and selecting an orthonormal mother wavelet ψ. In this case, the family $\Psi(\alpha, \beta)$ is furthermore an orthonormal basis.

3) Summary and Discussion: Referring to our discussion about the objectives of the paper, we may state the following at this point:

- **The "universal approximation" property** is guaranteed for finite sums of the form (5) if ψ is chosen as an appropriate wavelet according to the discussion above.
- **Explicit link between the network coefficients and the wavelet transform** is provided. To make this clearer for wavelets of the Morlet–Grossmann class, rewrite decomposition (8) in the form

$$f(\boldsymbol{x}) = \frac{1}{C_\psi}\int_{\Re^n}\int_{\Re_+^n}(\det D)^{\frac{1}{2}}\psi[D(\boldsymbol{x}-\boldsymbol{t})](\mathcal{W}(\boldsymbol{d},\boldsymbol{t})d\boldsymbol{d}d\boldsymbol{t})$$

Then an approximation of the form

$$f(\boldsymbol{x}) \approx \sum_{i=1}^{N} w_i \det(D_i^{\frac{1}{2}})\psi[D_i\boldsymbol{x}-\boldsymbol{t}_i]$$

can be thought of as an approximation of the measure $(\mathcal{W}(\boldsymbol{d},\boldsymbol{t})d\boldsymbol{d}d\boldsymbol{t})$ by some suitable linear combination of Dirac measures.[6] And it turns out that $\mathcal{W}$ can be roughly estimated from data using decomposition formula (9) directly.

- Referring to the two types of approximation (4) and (5), we shall call them the (truncated) *wavelet decomposition* and *wavelet network*, respectively. In the wavelet decomposition, only the weights w_k will be identified, while the dilation and translations will follow the regular grid structure. In contrast, in the wavelet network, weights, dilations, and translations will jointly be fitted from data. We shall provide in the sequel algorithms to learn the parameters of these two types of approximants using noisy input/output data.

[6]This is the argument used in [5] to establish the link between neural networks and the Radon transform, however this link fails to provide directly approximants in the form of neural networks.

C. Wavelet Networks and Their Parametrization

Based on the previous discussion, we propose a network structure of the form

$$g(x) = \sum_{i=1}^{N} w_i\psi[D_i(\boldsymbol{x}-\boldsymbol{t}_i)] + \bar{g} \quad (13)$$

where the additional (and redundant) parameter $\bar{g}$ is introduced to help dealing with nonzero mean functions on finite domains. Note that this form (13) is equivalent to the form (5) up to the constant $\bar{g}$ since the dilations and translations are adjustable. Furthermore, in order to compensate for the orientation selective nature of the dilations in (13), we combine a rotation with each affine transform to make the network more flexible. Our *wavelet network* structure is thus of the following form:

$$g(\boldsymbol{x}) = \sum_{i=1}^{N} w_i\psi[D_iR_i(\boldsymbol{x}-\boldsymbol{t}_i)] + \bar{g} \quad (14)$$

where

- the additional parameter $\bar{g}$ is introduced in order to make it easier to approximate functions with nonzero average, since the wavelet $\psi(\boldsymbol{x})$ is zero mean;
- the dilation matrices D_i's are diagonal matrices built from dilation vectors, while R_i's are rotation matrices.

This network structure is illustrated in Fig. 2.

III. Learning Algorithm

In this section we propose an algorithm for adjusting the parameters of our wavelet network. The learning is based on a sample of random input/output pairs $\{\boldsymbol{x}, f(\boldsymbol{x})\}$ where $f(\boldsymbol{x})$ is the function to be approximated. A stochastic gradient type algorithm will be introduced for this purpose, which is very similar to the backpropagation algorithm for neural network learning. More precisely, in the sequel we are given a sequence of random pairs $\{\boldsymbol{x}_k,\ y_k = f(\boldsymbol{x}_k) + v_k\}$ where $\{v_k\}$ is observation noise.

A. Principle of the Stochastic Gradient Algorithm

Collect all the parameters $\bar{g}$, w_i's, $\boldsymbol{t}_i$'s, D_i's, and R_i's in a vector $\boldsymbol{\theta}$ and write $g_\theta(\boldsymbol{x})$ to refer to the network defined by (14) with the parameter vector $\boldsymbol{\theta}$. The objective function to be minimized is

$$C(\boldsymbol{\theta}) = \frac{1}{2}\boldsymbol{E}\{[g_\theta(\boldsymbol{x}) - y]^2\}. \quad (15)$$

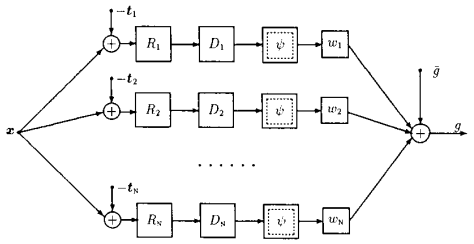


Fig. 2. Wavelet network structure for approximation. The combination of translation, rotation, dilation, and wavelet lying on the same line will be called a *wavelon* in the sequel.

Off-line gradient or Newton methods may be used, but the computation of the gradient and the Hessian matrix is generally very heavy. Hence we preferred to implement a stochastic gradient algorithm [15], [16] to recursively minimize the criterion (15) using input/output observations. This algorithm modifies the parameter vector $\boldsymbol{\theta}$ after each measurement $(\boldsymbol{x}_k, y_k)$ in the opposite direction of the gradient of the functional

$$c(\boldsymbol{\theta}, \boldsymbol{x}_k, y_k) = \frac{1}{2}[g_\theta(\boldsymbol{x}_k) - y_k]^2. \tag{16}$$

B. Handling the Dilation and Rotation Parameters

Up to now we have denoted dilation parameters by

$$D_i = \text{diag}(\boldsymbol{d}_i) = \text{diag}(d_i^1, \ldots, d_i^n)$$

where $\boldsymbol{d}_i$ measures the inverse of the scale of the concerned wavelet. Another possible choice is

$$D_i = \text{diag}\left(\frac{1}{s_i^1}, \ldots, \frac{1}{s_i^n}\right).$$

The related vector $\boldsymbol{s}_i = (s_i^1, \ldots, s_i^n)^T$ directly measures the scale of the concerned wavelet. If we compare the partial derivatives of the functional (16) w.r.t. $\boldsymbol{d}_i$ and $\boldsymbol{s}_i$,

$$\frac{\partial c}{\partial \boldsymbol{d}_i} = e_k w_i \text{diag}\,[R_i(\boldsymbol{x}_k - \boldsymbol{t}_i)]\boldsymbol{\psi}'[D_i R_i(\boldsymbol{x}_k - \boldsymbol{t}_i)] \tag{17}$$

$$\frac{\partial c}{\partial \boldsymbol{s}_i} = -e_k w_i D_i^2 \text{diag}[R_i(\boldsymbol{x}_k - \boldsymbol{t}_i)]\boldsymbol{\psi}'[D_i R_i(\boldsymbol{x}_k - \boldsymbol{t}_i)] \tag{18}$$

where $\boldsymbol{\psi}'(\boldsymbol{x}) = d\psi(\boldsymbol{x})/d\boldsymbol{x}$ and $e_k = g_\theta(\boldsymbol{x}_k) - y_k$, we observe that the partial derivatives in (18) are smaller for larger scales (larger $\boldsymbol{s}_i$'s), a nice property which is not satisfied by (17). In order to make the stochastic gradient learning algorithm more cautious for larger scale wavelets, the form (18) is preferred.

Since the learning of the rotation parameters is rather complicated, we propose two approaches: one using the Givens rotations and another using an iterative orthonormalization procedure.

Rotation matrices can be factorized using elementary Givens rotations with angles α_i^j's as follows (the index i is omitted for the sake of clarity):

$$R(\alpha^1, \ldots, \alpha^m) = \begin{bmatrix} \cos\alpha^1 & -\sin\alpha^1 & & & & \\ \sin\alpha^1 & \cos\alpha^1 & & & & \\ & & 1 & & 0 & \\ & & & \ddots & & \\ & 0 & & & \ddots & \\ & & & & & 1 \end{bmatrix} \begin{bmatrix} \cos\alpha^2 & 0 & -\sin\alpha^2 & & & \\ 0 & 1 & 0 & & & \\ \sin\alpha^2 & 0 & \cos\alpha^2 & & 0 & \\ & & & 1 & & \\ & 0 & & & \ddots & \\ & & & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & \ddots & & 0 & \\ & & & 1 & & \\ & 0 & & & \cos\alpha^m & -\sin\alpha^m \\ & & & & \sin\alpha^m & \cos\alpha^m \end{bmatrix}$$

$$\text{where} \quad m = \frac{1}{2}n(n-1) \tag{19}$$

Thus Givens rotations must be identified, which is feasible only for problems of low dimension. For problems of larger dimension ($n > 3$), we process the rotation matrices in a different way.

For this alternative approach, at the beginning of each step of the stochastic gradient procedure, the R matrices are assumed to be rotation matrices. Then the entries of these rotation matrices are modified in the opposite of the (stochastic) gradient. After such a modification, these matrices are generally no longer rotation matrices. So we project them into the set of rotation matrices by solving the following approximation problem: given a square nonsingular matrix $\tilde{R} \in \Re^{n\times n}$, find

$$R = \arg\min_{R\in\mathcal{R}} \|R - \tilde{R}\|_\infty \text{ with}$$

$$\mathcal{R} = \{R:\ R \in \Re^{n\times n},\ R^T R = I,\ \det R = 1\}.$$

For this purpose, a symmetric orthonormalization algorithm introduced in [17] is used. This is an iterative procedure which converges very rapidly when the matrix to be orthonormalized is close to an orthonormal one (this happens to be the case for us since only a small modification of the entries is performed at the stochastic gradient stage). In Appendix B, a brief description of this algorithm is provided.

C. Calculating the Stochastic Gradient

Explicit formulae for the partial derivatives of the functional (16) w.r.t. each component of the parameter vector $\boldsymbol{\theta}$ are now listed. From these formulae the calculation of the gradient follows immediately. For convenience, we shall use the following notations:

$$\boldsymbol{\psi}'(\boldsymbol{x}) = d\psi(\boldsymbol{x})/d\boldsymbol{x},\ \ e_k = g_\theta(\boldsymbol{x}_k) - y_k \text{ and}$$
$$\boldsymbol{z}_i = D_i R_i(\boldsymbol{x}_k - \boldsymbol{t}_i).$$

The partial derivatives of the functional $c(\boldsymbol{\theta}, \boldsymbol{x}_k, y_k)$ w.r.t. $\bar{g}$, w_i, $\boldsymbol{t}_i$, $\boldsymbol{s}_i$ and R_i are respectively given by

$$\begin{aligned}
\frac{\partial c}{\partial \bar{g}} &= e_k \\
\frac{\partial c}{\partial w_i} &= e_k \psi(\boldsymbol{z}_i) \\
\frac{\partial c}{\partial \boldsymbol{t}_i} &= -e_k w_i R_i^T D_i \boldsymbol{\psi}'(\boldsymbol{z}_i) \\
\frac{\partial c}{\partial \boldsymbol{s}_i} &= -e_k w_i D_i^2 \mathrm{diag}[R_i(\boldsymbol{x}_k - \boldsymbol{t}_i)] \boldsymbol{\psi}'(\boldsymbol{z}_i) \\
\frac{\partial c}{\partial R_i} &= e_k w_k D_i \boldsymbol{\psi}'(\boldsymbol{z}_i)(\boldsymbol{x}_k - \boldsymbol{t}_i)^T.
\end{aligned}$$

At this point we are ready to implement the stochastic gradient procedure, additional work is required, however. As is the case for the backpropagation algorithm for neural network learning [18,19], the objective function (15) is likely to be highly nonconvex, so local minima are expected. To improve the situation, careful initialization of the algorithm is performed and appropriate constraints are set on the adjusted parameters. While no theoretical investigation is available which may guarantee convergence, drastic improvement was exhibited: the original stochastic gradient usually diverged but our modified algorithm always performed successfully in our experiments. As far as we know, a similar situation is encountered in feedforward neural network training. The corresponding additional features are now presented.

D. Setting Constraints on the Adjustable Parameters

Let $f : \mathcal{D} \to \Re$ be the function to be approximated, where $\mathcal{D} \subset \Re^n$ is the domain where the approximation should be made. The following constraints on the parameters are introduced:

1) To keep the wavelets inside or near to the domain $\mathcal{D}$, select another domain $\mathcal{E}$ such that $\mathcal{D} \subset \mathcal{E} \subset \Re^n$, and require

$$\boldsymbol{t}_i \in \mathcal{E}, \quad i = 1, 2, \ldots, N. \tag{20}$$

2) To avoid excessive compression of each wavelet, select $\varepsilon > 0$ and require

$$D_i^{-1} > \varepsilon I, \quad i = 1, 2, \ldots, N. \tag{21}$$

3) To prevent the total volume of the wavelet supports from being too small, select a $V > 0$ and require

$$\sum_{i=1}^{N} (\det D_i)^{-1} > V. \tag{22}$$

4) The rotation matrices should verify

$$R_i^T R_i = I \text{ and } \det R_i = 1, \quad i = 1, 2, \ldots, N. \tag{23}$$

Note that the wavelets we use are not necessarily compactly supported, but are always rapidly vanishing, so the notion of "support" used in the third constraint should be taken in an approximate sense. A formal definition for the notion of support in such a case was given by Y. C. Pati and P. S. Krishnaprasad in [20]. To satisfy these constraints, a *projected* stochastic gradient procedure is implemented as depicted in Fig. 3.

E. Network Initialization

This is where the link to wavelet decomposition can be used. More precisely, an initial guess for the network parameters can be derived by using the *decomposition* formula (9) (recall that in contrast our network structure mimics the *synthesis* formula (8)). The idea is the following: using noisy input/output measurements $\{\boldsymbol{x}, f(\boldsymbol{x})\}$, we can get some rough estimate of $\mathcal{W}(h, \boldsymbol{t})$ in (9) by replacing the integration by an appropriate averaging of the observations. In this paper, we only propose simple heuristic implementations of this idea.

1) The One-Dimensional Case: For the sake of simplicity, we consider first the one dimensional case. Assume that we want to approximate the function $f(x)$ over the domain $\mathcal{D} = [a, b]$ by a network of the form

$$g(x) = \sum_{i=1}^{N} w_i \psi\left(\frac{x - t_i}{s_i}\right) + \bar{g}.$$

Note that no rotation parameter is needed for one dimensional case. The initialization of this wavelet network consists in the evaluation of the parameters $\bar{g}$, w_i, t_i and s_i for $i = 1, 2, \ldots, N$. To initialize $\bar{g}$ we need to estimate the mean of the function $f(x)$ (from its available observations) and set $\bar{g}$ to this estimated mean. w_i's are simply set to zero. The rest of the problem is how to initialize t_i's and s_i's.

To initialize t_1 and s_1, select a point p between a and b: $a < p < b$. The choice of this point will be detailed later. Then we set

$$t_1 = p, \quad s_1 = \xi(b - a),$$

where $\xi > 0$ is a properly selected constant (the typical value of ξ is 0.5). The interval $[a, b]$ is divided into two parts by the point p. In each sub-interval, we recursively repeat the same procedure which will initialize t_2, s_2 and t_3, s_3, and so on, until all the wavelets are initialized. This procedure applies in this form when a number of wavelons is used which is a power of 2. When this is not the case, the recursive procedure is applied as long as possible, then the remaining (t_i) are initialized at random for the finest remaining scale.

The following method is proposed to select the point p inside $[a, b]$ (and recursively points which divide all the sub-intervals). Introduce a "density" function

$$\rho(x) = \frac{\varrho(x)}{\int_a^b \varrho(x) dx}, \text{ where } \varrho(x) = \left|\frac{df(x)}{dx}\right|$$

must be estimated from noisy input/output observations $\{x, f(x)\}$. A very rough estimate of $\varrho(x)$ is used for this

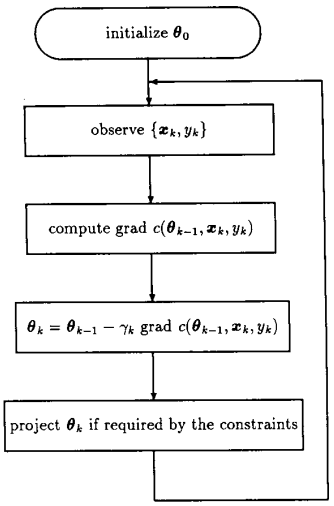


Fig. 3. The learning algorithm.

purpose. Then we take the point p to be the center of gravity of $[a, b]$

$$p = \int_a^b x\rho(x)dx.$$

2) The Multidimensional Case: For the multidimensional case, the initialization of $\bar{g}$ and the w_i's is performed in the same way as for one-dimensional case: $\bar{g}$ is set to the estimated mean of the function to be approximated, and the w_i's are set to zero. The rotations are initialized to identity. To initialize $\boldsymbol{t}_i$'s and $\boldsymbol{s}_i$'s, we handle the different coordinates separately using the procedure for the scalar case, and then we take all the possible combinations between these components to initialize the $\boldsymbol{t}_i$'s and the $\boldsymbol{s}_i$'s.

F. Miscellaneous

Model Order Selection: The selection of the number of wavelons in the network may be performed by relying on appropriate versions of standard model order criteria in statistics, e.g., Akaike criteria or Rissanen's Minimum Description Length principle [21]. In all cases, such an approach amounts to adding to the objective function an additional term which penalizes the number of adjusted parameters, following the principle of parsimony. At this stage of our study, we have not yet designed an appropriate *automatic* procedure for adjusting the number of wavelons directly from data, but our preliminary experience was that good robustness is achieved by our algorithm with respect to this issue.

Normalizing the Domain of Approximation: A simple additional processing is implemented in order to recover in a very elementary way some of the "normalizing" features that are usually provided by Newton-type methods. The domain $\mathcal{D}$ of the $\boldsymbol{x}$ data (which we take to be a rectangle) is transformed into the hypercube $[-1, 1]^n$. The learning procedure is applied on this hypercube, and the desired approximation is recovered by transforming $\boldsymbol{x}$ back to its original shape.

IV. Experimental Results

In this section, results on simulated as well as real data are reported. For the simulations we use records of randomly generated $\{\boldsymbol{x}_k\}$'s that are independent and uniformly distributed on the considered domain $\mathcal{D}$. Then we take $y_k = f(\boldsymbol{x}_k) + v_k$ where (v_k) is a simulated white noise independent from the $\boldsymbol{x}_k$'s, with standard deviation being approximately 10% of the size of the range of the function f. When real data records $\{\boldsymbol{x}_k, y_k\}$ are processed, they are not processed on-line, but are read from a file by selecting their index k at random with uniform distribution in the set $\{1, \ldots, K\}$ where K is the number of observations.

To assess the approximation results, a *figure of merit* is needed. An obvious candidate for this purpose is the empirical estimate of the least squares objective function itself, namely $\sum_{k=1}^{K} \frac{1}{2}[g_\theta(\boldsymbol{x}_k) - y_k]^2$. However this figure of merit provides a biased assessment by favoring the areas where $\boldsymbol{x}$'s are frequently selected: this may for instance be the case for real experiments, where the actual distribution of $\boldsymbol{x}$ is not chosen. Since we are also interested in assessing the extrapolation ability of our procedure, we preferred to use the following alternative figure of merit. A set of test points $\mathcal{T} = \{\boldsymbol{x}_j : j = 1, 2, \ldots, M\} \subset \mathcal{D}$ is selected and we assume that $y_j = f(\boldsymbol{x}_j) + v_j$ is available for $j = 1, 2, \ldots, M$. Then we take

$$\delta(g_\theta) = \sqrt{\frac{\sum_{j=1}^{M}[g_\theta(\boldsymbol{x}_j) - y_j]^2}{\sum_{j=1}^{M}(y_j - \bar{y})^2}} \quad \text{with} \quad \bar{y} = \frac{1}{M}\sum_{j=1}^{M} y_j$$

as a figure of merit.

We have compared our wavelet network, the neural network and the wavelet decomposition (i.e., with fixed dilations and translations) on the same examples. The tested neural network follows Cybenko's structure with the sigmoidal function

$$\sigma(x) = \frac{1 - e^{-x}}{1 + e^{-x}}.$$

When learning the corresponding neural network using a backpropagation algorithm, we use the parametrization $\sigma(\tilde{\boldsymbol{a}}^T(\boldsymbol{x} - \tilde{\boldsymbol{b}}))$ for each neuron instead of $\sigma(\boldsymbol{a}^T\boldsymbol{x} + b)$, the reason for this is that the partial derivative

$$\frac{\partial}{\partial \boldsymbol{a}}\sigma(\boldsymbol{a}^T\boldsymbol{x} + b) = \sigma'(\boldsymbol{a}^T\boldsymbol{x} + b)\boldsymbol{x}$$

depends on the value of $\boldsymbol{x}$, while in contrast

$$\frac{\partial}{\partial \tilde{\boldsymbol{a}}}\sigma(\tilde{\boldsymbol{a}}^T(\boldsymbol{x} - \tilde{\boldsymbol{b}})) = \sigma'(\tilde{\boldsymbol{a}}^T(\boldsymbol{x} - \tilde{\boldsymbol{b}}))(\boldsymbol{x} - \tilde{\boldsymbol{b}})$$

depends on the value of $(\boldsymbol{x} - \tilde{\boldsymbol{b}})$. This latter choice makes the backpropagation algorithm more stable. After the learning procedure, we reset $\boldsymbol{a} = \tilde{\boldsymbol{a}}$ and $b = \tilde{\boldsymbol{a}}^T\tilde{\boldsymbol{b}}$.

The wavelet decomposition (cf. (4)) is estimated from available data with the help of a standard least squares algorithm for minimizing the objective function

$$\sum_k \left[f(\boldsymbol{x}_k) - \left(\sum_{i=1}^{N} w_i \varphi_i(\boldsymbol{x}_k) \right) \right]^2$$

with respect to the weights w_i.

To facilitate the terminology, we use the common term *unit* to refer to neurons and wavelons in the following. Furthermore, to make comparisons as fair as possible as far as the number of adjusted parameters is concerned, we have adopted the following procedure: for each experiment, the number of units in each type of network is selected in such a way that the resulting numbers of adjusted parameters be as close as possible to each other. Moreover, in any case, the number of adjusted parameters has been always taken to be the least one for the wavelet network. The networks and their learning algorithms have been implemented in C language on a Sun4 Sparc-station.

1) Approximation of a Single Variable Function: Simulation Results: The wavelet function we have taken is the so-called "Gaussian-derivative" $\psi(x) = -xe^{-\frac{1}{2}x^2}$ to which the general approximation theorem described in Section II-B applies. Fig. 4 shows the results for the approximation of the piecewise defined function

$$f(x) = \begin{cases} -2.186x - 12.864 & \text{if } -10 \le x < -2 \\ 4.246x & \text{if } -2 \le x < 0 \\ 10e^{-0.05x-0.5} & \\ \quad \cdot \sin[(0.03x + 0.7)x] & \text{if } 0 \le x \le 10. \end{cases} \tag{24}$$

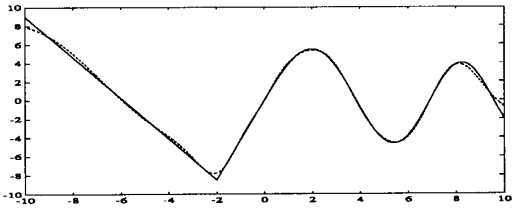
over the domain $\mathcal{D} = [-10, 10]$. For comparison we show the results with our wavelet network, with the sigmoid-based neural network, and with the wavelet decomposition. The solid lines represent the function f and the dashed lines show the approximations. We can see that the wavelet network gives the best approximations for the smooth segment as well as for the sharp segment. The figure of merit δ for each approximation is computed on a uniformly sampled test set of 200 points. Comparable results have been found for a number of wavelons ranging from 5 to 11 for our wavelet network.

2) Approximation of Two Variable Functions: Simulations and a Real Experiment: In both cases, we have selected $\psi(\boldsymbol{x}) = x_1 x_2 e^{-\frac{1}{2}(x_1^2+x_2^2)}$ as our wavelet function.

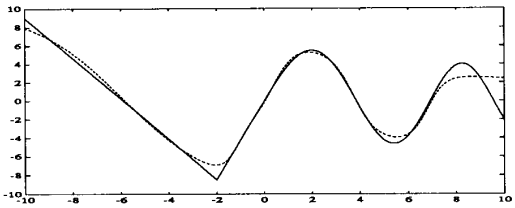
The first example is a simulation study to approximate the function $f(\boldsymbol{x}) = (x_1^2 - x_2^2)\sin(0.5x_1)$ over the domain $\mathcal{D} = [-10, 10] \times [-10, 10]$. Table I shows the approximation results using the three methods. The figure of merit δ for each method is computed on a uniformly sampled test set of 400 points. Fig. 5 illustrates the form of $f(\boldsymbol{x})$ over $\mathcal{D}$ and its approximation realized by the wavelet network.

For another example, real data were taken from a gas turbine compressor.[7] Our task is to identify the function $\eta = f(\pi, \omega)$ where η, π and ω are respectively the thermodynamical efficiency, the output/input pressure ratio and the corrected rotation velocity. The form of this function is illustrated by Fig. 6 in which we can see that the observations were irregularly sampled in several slices between which no observation is available (shown by plane zones in the figure). The networks are learned with 2000 measurement points and the results are tested on 3294 points to compute δ. Table II shows the approximation results obtained with the three methods. Fig. 6 shows also the resulting approximation for the wavelet network. Note that in the white zone no observation is available, so the result in this zone is not significant. In this experiment, the domain of operation of the plant was taken too small to really exhibit the nonlinearities expected by the process engineer.

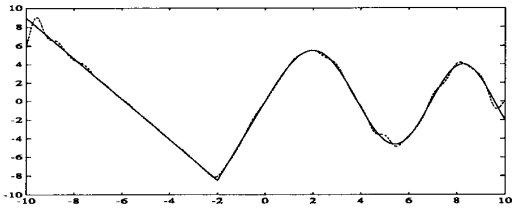
[7] With the courtesy of European Gas Turbine SA.



wavelet network, 7 wavelons, 22 parameters,
10000 learning iterations, $\delta = 0.05057$.



neural network, 7 neurons, 22 parameters,
10000 learning iterations, $\delta = 0.13286$



wavelet decomposition, 31 wavelons, 32 parameters,
10000 learning iterations, $\delta = 0.06350$

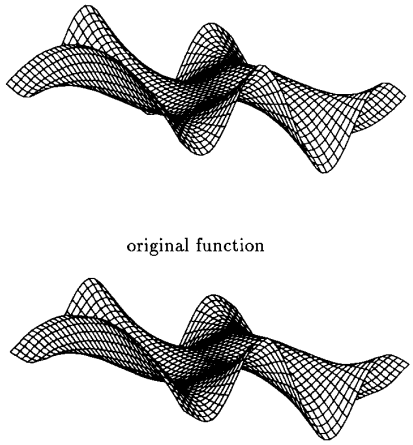
Fig. 4. Approximation results of function (24).

TABLE I
APPROXIMATION RESULTS OF $f(\boldsymbol{x}) = (x_1^2 - x_2^2)\sin(0.5x_1)$

method	number of units	number of parameters	number of iterations	δ
wavelet network	49	442	40000	0.03395
neural network	225	1126	40000	0.29381
wavelet decomposition	961	962	40000	0.14289

V. CONCLUSION AND COMMENTS

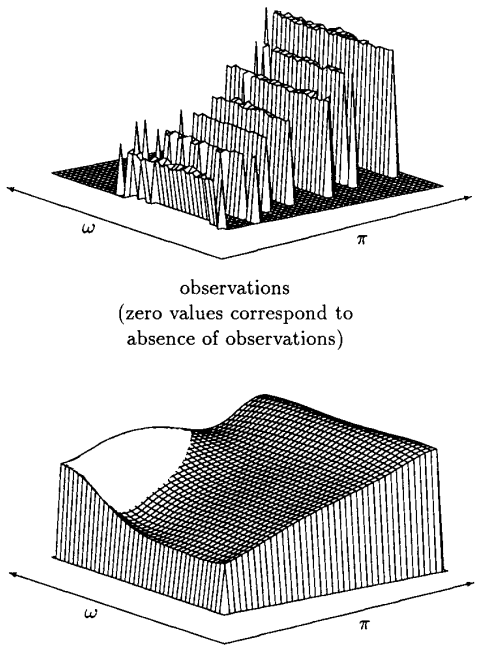
In this paper a new method has been introduced for identifying general nonlinear static systems from input/output observations. This method was inspired by both neural networks and the wavelet decomposition as introduced by Grossmann-Morlet and further studied by Y. Meyer, I. Daubechies, and many others. The basic idea is to replace the neurons by "wavelons," i.e., computing units obtained by cascading an



original function

resulting approximation

Fig. 5. Approximation of the function $f(\boldsymbol{x}) = (x_1^2 - x_2^2)\sin(0.5x_1)$ by a wavelet network over $\mathcal{D} = [-10, 10] \times [-10, 10]$.



observations
(zero values correspond to
absence of observations)

resulting approximation
(no observation was available
in the white zone)

Fig. 6. Approximation of the function $\eta = f(\pi, \omega)$ by wavelet network.

affine transform and a multidimensional wavelet. Then these affine transforms and the "synaptic weights" have to be identified from possibly noise corrupted input/output data.

In performing this study, we had several objectives in mind, that have been presented at the end of Section II-A. Let us discuss how far these objectives have been achieved.

TABLE II
APPROXIMATION RESULTS OF $\eta = f(\pi, \omega)$

method	number of units	number of parameters	number of iterations	δ
wavelet network	9	82	10^5	0.06041
neural network	16	81	10^5	0.09240
wavelet decomposition	81	82	10^5	0.08311

1) **Guarantee the "universal approximation" property:** this has been discussed in Section II-B and is a direct byproduct of the wavelet decomposition.
2) **Have an explicit link between the network coefficients and some appropriate transform:** this is also automatically provided by the wavelet decomposition. We have used this in two different ways. First, the intuitive nature of the notions of translation, scale, and rotation, helped us to design the constraints in an appropriate way. Second, the availability of a *direct* and closed form formula for computing the (continuous) wavelet transform was useful for designing an initial guess of the wavelet network coefficients. Both additional features of our "backpropagation" algorithm helped drastically to improve its behavior. Using the mentioned explicit link in a more fundamental way is under progress and will be reported elsewhere.
3) **Possibly achieve the same quality of approximation with a network of reduced size.** Reported results would favor this claim, but it is wise at this point to be somewhat cautious if the complexity is measured in terms of the number of adjusted parameters, since the neural network procedure we have used for comparison might be improved as well. On the other hand, the wavelet network seems more efficient than the (fitted) wavelet decomposition, although the latter one is likely to be more robust however since it is just a linear regression algorithm. Finally, it is worth noticing that the number of *units* is generally smaller in wavelet networks than in neural networks, especially for problems of higher dimension. And it turns out that, for the particular wavelet we have chosen, the cost of implementing the nonlinearity (sigmoid versus Gaussian—derivative) is mainly proportional to the number of units.

Our wavelet network has been used for black-box identification of nonlinear static systems. Extension to nonlinear dynamical systems of the form

$$y_k = g_\theta(y_{k-1}, y_{k-2}, \ldots, y_{k-l}, \boldsymbol{u}_k) + v_k$$

or in state space form is conceptually straightforward, but several questions arise. First, the requested number of wavelons drastically increases with the model order l. Second, no system theory is available for "dynamical wavelet networks" which is certainly a drawback. Further experiments must be performed in this direction before understanding the range of validity of such a black-box identification method for general nonlinear dynamical systems.

APPENDIX

From One- to Multidimensional Continuous Wavelets

We first derive formulae (8, 9). These formulae are known for $n = 1$, see [9]. They generalize immediately for functions f of the form

$$f(\boldsymbol{x}) = f_1(x_1) \times \ldots \times f_n(x_n) \tag{25}$$

since each component in the integral is handled separately. By linearity, (8, 9) extend to linear combinations of such f's, i.e., to a dense subset of $\mathcal{L}^2(\Re^n)$. But the map $f \mapsto (\det D)^{\frac{1}{2}} \mathcal{W}(\boldsymbol{d}, \boldsymbol{t})$ defined by (9) is a bounded linear operator from $\mathcal{L}^2(\Re^n)$ into $\mathcal{L}^2(\Re^n \times \Re_+^n, d\boldsymbol{t} \times \det D.d\boldsymbol{d})$ (the direct product of scalar affine groups with their right-Haar measures). Then the composition of maps $f \mapsto (\det D)^{\frac{1}{2}} \mathcal{W}(\boldsymbol{d}, \boldsymbol{t})$ defined by (9) and $(\det D)^{\frac{1}{2}} \mathcal{W}(\boldsymbol{d}, \boldsymbol{t}) \mapsto g$ defined by (8) yields $g = f$ for f in a dense subset of $\mathcal{L}^2(\Re^n)$, hence for all $f \in \mathcal{L}^2(\Re^n)$. Consequently formulae (8,9) extend to $\mathcal{L}^2(\Re^n)$.

We prove next that family (11) is a frame. Again this result is known for $n = 1$ [9], and we denote by Φ_s the corresponding frame. Denote by $c_{\min}$ and $c_{\max}$ the frame constants associated with the scalar wavelet ψ_s. Taking f of the form (25), we have on the one hand

$$\|f\|^2 = \|f_1\|^2 \times \ldots \times \|f_n\|^2$$

and on the other hand

$$\sum_{\phi \in \Phi} |< \phi, f >|^2 = \sum_{\phi_1 \in \Phi_s} \cdots \sum_{\phi_n \in \Phi_s} \prod_{i=1}^{n} |< \phi_i, f_i >|^2 = \prod_{i=1}^{n} \left(\sum_{\phi_i \in \Phi_s} |< \phi_i, f_i >|^2 \right).$$

Hence frame inequalities (3) hold for such f's with the family $\Phi = \Psi_s(\alpha, \beta)$ as defined in (10) with the same frame constants $c_{\min}$ and $c_{\max}$. Next consider a linear combination

$$f = \sum_k \lambda_k f_k \tag{26}$$

where the f_k take the form (25) and are *mutually orthogonal;* note that such linear combinations are dense in $\mathcal{L}^2(\Re^n)$. For f of the form (26), we have

$$\sum_{\phi \in \Phi} |< \phi, f >|^2 = \sum_k |\lambda_k|^2 \sum_{\phi \in \Phi} |< \phi, f_k >|^2$$

$$\|f\|^2 = \sum_k |\lambda_k|^2 \|f_k\|^2$$

so that inequalities (3) extend to such f's. Finally, that $\Psi_s(\alpha, \beta)$ is a frame follows by density.

Symmetric Orthogonalization of Matrices

In this appendix we describe the algorithm of symmetric orthogonalization of matrices. For more details and the proofs see [17]. Our purpose is to orthogonalize the matrix

$$A \in \mathcal{C}^{n \times p} \text{ with } \operatorname{rank}(A) = p \leq n.$$

where $\mathcal{C}$ denotes the complex numbers. For this we compute $T = S^{-\frac{1}{2}}$ where $S = A^*A$, A^* denotes the adjoint of A, and $S^{-\frac{1}{2}}$ is the symmetric inverse square root of S. It is easy to see that $R = AT$ is orthogonal. Furthermore, it is proved in [17] that the matrix R computed in this way satisfies

$$\|A - R\| = \min_{Q \in \mathcal{U}} \|A - Q\|$$

where $\mathcal{U}$ is the set of all orthogonal matrices of $\mathcal{C}^{n \times p}$. This result is true for both the Euclidean norm and the Frobenius norm. In this sense we say that R is the symmetric orthogonalization of A.

Let $\rho(S)$ be the spectral radius of S; it is shown in [17] that if $\mu < \sqrt{3/\rho(S)}$, then the scheme

$$T_0 = \mu I$$

$$T_{m+1} = T_m + \frac{1}{2} T_m (I - T_m S T_m)$$

quadratically converges to $S^{-\frac{1}{2}}$. Moreover, if $K(S) < 9$ where $K(S)$ is the ratio of the extremal eigenvalues of S, then this scheme is stable w.r.t. rounding errors. This condition can be weakened into $K(S) < (17 + 6\sqrt{8})$ if a symmetrization is performed at every step on T_m.

In order to define an initial guess, the spectral radius $\rho(S)$ of the matrix S has to be estimated. In fact, the ∞-norm is used instead of this spectral radius. It is shown that

$$\mu = \sqrt{\frac{3}{\|S\|_\infty}} \leq \sqrt{\frac{3}{\rho(S)}}$$

so the initialization $T_0 = \mu I$ will ensure the convergence of the algorithm.

The first iteration can be skipped since it is easy to compute the following:

$$T_1 = \frac{3}{2} \mu I - \frac{1}{2} \mu^3 S.$$

However, if the matrix $\Delta = S - I$ is small (i.e., $\rho(\Delta) < 1$), the initial guess can be much improved by taking for T_0 the Taylor expansion of order k of $(I + \Delta)^{-\frac{1}{2}}$

$$T_0 = I + \sum_{i=1}^{k} (-1)^i \binom{-\frac{1}{2}}{i} \Delta^i.$$

After m iterations the magnitude of the error is given by

$$T_m - S^{-\frac{1}{2}} = \boldsymbol{o}(\Delta^{(k+1)2^m}).$$

Symmetrization of T_m is performed at every stage to improve the stability when needed. The algorithm is now given:

begin
  $S := A^*A$;
  $\Delta := S - I$;
  $\delta := \|\Delta\|_\infty$;
  if $(\delta < \varepsilon)$ = **then**
    nothing to do;
  elseif $(\delta < 1)$ = **then**
    $k =:$ Taylor approximation order;
    $T =:$ Taylor approximation of order k;
    sym := false;

```
        else
            μ := √(3/δ);
            T := (3/2)μI − (1/2)μ³S;
            sym := true;
    endif;
    iter := 0;
        loop:
            δ0 := δ;
            Z := I − T × S × T;
            δ := ‖Z‖∞;
            if (δ < ε) then exit of the loop endif;
            if (δ > δ0) then divergence endif;
            iter := iter +1;
            T := (1/2)T × (2I + Z);
            if (sym) then T := (1/2)(T* + T) endif;
        endloop;
    R = A × T;
end.
```

REFERENCES

[1] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proc. IEEE*, vol. 78, no. 9, Sept. 1990.

[2] K. Hornik, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, 1989.

[3] G. Cybenko, "Approximation by superposition of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, pp. 303–314, 1989.

[4] A. R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," Tech. Rep. 58,Dept. of Statistics, Univ. of Illinois at Urbana-Champaign, 1991, to appear in *IEEE Trans. Informat. Theory*.

[5] S. M. Carrol and B. W. Dickinson, "Construction of neural nets using the Radon transform," in *Proc. IJCNN*, 1989.

[6] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Proc. IJCNN*, Washington, DC, June 18–22, 1989, I-593.

[7] L. K. Jones, "Constructive approximations for neural networks by sigmoidal functions," *Proc. IEEE*, vol. 78, Oct. 1990.

[8] I. Daubechies, "Orthonormal bases of compactly supported wavelets," *Communications on Pure and Applied Math.*, vol. 91, pp. 909–996, 1988.

[9] I. Daubechies, "The wavelet transform, time-frequency localization and signal analysis," *IEEE Trans. Informat. Theory*, vol. 36, Sept. 1990.

[10] S. G. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Trans. PAMI*, Vol. 11 no. 7, July 1989.

[11] S.G. Mallat, "Multiresolution approximation and wavelets orthonormal bases of $\mathcal{L}^2(\Re)$", *Trans. Amer. Math. Soc.*, 315, no. 1 , 69–88, 1989.

[12] Y. Meyer, "Wavelets and operators", *Proceedings of the Special year in modern Analysis*, Urbana 1986/87, published by Cambridge University Press, 1989. See also Y. Meyer, *Ondelettes et Opérateurs*, Hermann, Paris, 1990.

[13] I. Daubechies, A. Grossmann and Y. Meyer, "Painless nonorthogonal expansions", *J. Math. Phys.*, vol. 27, p. 1271—1283, 1986.

[14] A. Grossmann and J. Morlet, "Decomposition of Hardy functions into square integrable wavelets of constant shape", *SIAM J. Math. Anal.*, vol. 15, pp. 723—736, 1984.

[15] A. Benveniste, M. Metivier and P. Priouret, *Adaptive Algorithms and Stochastic Approximations*, Applications of Mathematics Vol. 22, Springer Verlag, 1990.

[16] L. Ljung and T. Söderström, *Theory and practice of recursive identification, MIT Press* 1983.

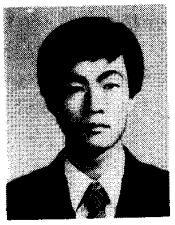
[17] B. Philippe, "An algorithm to improve nearly orthonormal sets of vectors on a vector processor", *SIAM J. ALG. DISC. METH.* vol. 8, no. 3, July 1987.

[18] P. Burrascano and P. Lucci, "A learning rule eliminating local minima in multilayer perceptrons", *Proc. ICASSP 90*, Albuquerque, Apr. 3–6, 1990.

[19] R. Hecht-Nielsen, "Back propagation error surfaces can have local minima", *IEEE-INNS Int. joint conference on neural networks*, Washington D.C., June 1989.

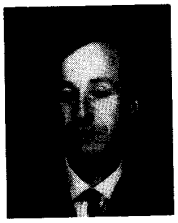
[20] Y. C. Pati and P. S. Krishnaprasad, "Analysis and synthesis of feedforward neural networks using discrete affine wavelet transformations", *Technical research report of the University of Maryland*, TR 90–44.

[21] E. J. Hannan, M. Deistler, *The Statistical Theory of Linear Systems*, Wiley Series in Probability and Mathematical Statistics, Wiley, 1988.



Qinghua Zhang was born in Shenyang, China, in 1963. He graduated from the University of Sciences and Techniques of China in 1986, and received the diploma of D.E.A. from Université de Lille I, France in 1988. From 1988 to 1991 he studied *industrial process monitoring* at IRISA, Rennes, and received the degree of "docteur de l'Université de Rennes I" in 1991.

Since 1992 he has been with the Department of Electrical Engineering at Linköping University, Sweden, as a guest researcher. His research interests are *failure detection and neural networks.*



Albert Benveniste (M'81, SM'89, F'91) was born on May 8, 1949 in Paris, France. In 1971 he graduated from Ecole des Mines de Paris. He performed the Thèse d'Etat in Mathematics, probability theory, in 1975.

From 1976 to 1979 he was Associate Professor of mathematics at the Université de Rennes I. From 1979 to present he has been Directeur de Recherche at INRIA. For his thesis, he worked on probability theory, stochastic processes, and ergodic theory. In parallel, he pursued work on automatic control and signal processing in the area of adaptive systems for time-varying systems, with applications to data communications. In 1980 he began joint work with Michèle Basseville on the subject of change detection in signals and dynamical systems with application to vibration mechanics and fault detection in process control. Since 1987 he has been involved in developing jointly with A. S. Willsky from MIT, a statistical theory of multiresolution signal and image processing, based on dynamical systems and Gaussian random fields on homogeneous trees. Since 1981 he has been interested in computer science, in the area of real-time languages and systems. Cooperating with P. LeGuernic, he participated to the definition of and theoretic studies on the Signal language.

Dr. Benveniste was cowinner of the IEEE TRANSACTIONS ON AUTOMATIC CONTROL Best Transaction Paper Award for his paper on blind deconvolution in data communications in 1980. In 1990 he received the CNRS silver medal. From 1986 to 1990 he was vice-chairman of the IFAC Committee on Theory and is now chairman of this committee for 1991–1993. From 1987 to 1990 he was Associate Editor for the IEEE TRANSACTIONS ON AUTOMATIC CONTROL. Currently he is Associate Editor for *International Journal of Adaptive Control and Signal Processing*, and the *International Journal of Discrete Event Dynamical Systems*. He is the Associate Editor at large for the IEEE TRANSACTIONS ON AUTOMATIC CONTROL. He has coauthored with M. Métivier and P. Priouret the book *Adaptive Algorithms and Stochastic Approximations*, and has been an editor, jointly with Michèle Basseville of the collective monograph *Detection of Abrupt Changes in Signals and Systems*.