Pergamon

PII: S0097-8485(98)00008-4

Computers Chem. Vol. 22, No. 5, pp. 355–361, 1998

Printed in Great Britain
0097-8485/98 $19.00 + 0.00

Neural Modeling of Torsional Potential Hypersurfaces in Non-rigid Molecules

Camelia Muñoz-Caro* and Alfonso Niño

Grupo de Química Computacional, E.U. Informática de Ciudad Real, Universidad de Castilla-La Mancha, Ronda de Calatrava 5, 13071, Ciudad Real, Spain

(Received 3 February 1998; Revised 19 March 1998)

Abstract—This paper presents a study on the ability of neural networks to model torsional potential hypersurfaces in non-rigid molecules. Using the potential function for the methyl torsion of acetaldehyde, we find that standard models do not represent accurately periodic function in its full range of definition. However, the functions are correctly described in any non-periodic zone. This behaviour arises from the periodic nature of the actual function and from the local character of the training methods. A new periodic activation function is defined, which enhances greatly the results for the periodic and non-periodic cases. The new activation function permits description of any periodic function of any number of arguments using a neural network with only one hidden layer. © 1998 Elsevier Science Ltd. All rights reserved.

Key words: neural networks, non-rigid molecules, potential hypersurfaces, activation function

1. INTRODUCTION

Neural networks are parallel distributed processing systems consisting of interconnected processing units usually called neurons. Each unit receives one or more input signals, generating a net input or activation value. This activation value generates an output by using a function (activation or transfer function). The output value is directed to other units through links affected by a weight coefficient. In the most common case, the nets are organized in several layers. Artificial neural networks appear in 1943 in the work of McCulloch and Pitts (McCulloch and Pitts, 1943). In this initial model only binary information was produced and transmitted. Around 1960, Rosenblatt and collaborators built the perceptron (Rojas, 1996). This neural network model consists in two layers of processing units (neurons) and it was considered a simplified model of the biological mechanisms of processing sensory information. However, a few years later Minsky and Papert (Minsky and Papert, 1969) found some intrinsic limitations of the perceptron, exemplified in the failure to represent the XOR ("or" exclusive) problem. Introduction of new non-linear activation functions and multilayered networks overcame these problems. Thus, in 1982 appeared the auto-associative Hopfield network (Hopfield, 1982) and we arrived at the widespread use of neural networks today.

From a mathematical point of view, neural networks are connectionist models that approximate functions based on Kolmogorov's theorem, or its more recent Sprecher's improvement (Sprecher, 1965). The theorem is a negative solution of Hilbert's thirteenth problem and states that any continuous function of any number of arguments can be exactly represented using a finite composition of functions of a single argument and addition. Also, it has been shown (Hecht-Nielsen, 1987; Hornik *et al.*, 1989; Kreinovich, 1991) that standard multilayer feedforward networks, using arbitrary nonlinear activation functions, can approximate any mapping, f: $\Re^m \rightarrow \Re^n$, with arbitrary precision. However, it is not determined which is the "best" activation function to use for a given problem and how many units are needed to attain given precision. In practical cases, this is usually considered a matter of trial and error.

Neural networks are useful modelization and mapping tools in the chemical field (Peterson, 1990; Bulsari and Saxén, 1991; Zupan and Gasteiger, 1993; Fisher et al, 1995; Kireev, 1995; So and Karplus, 1996; Ivanciuc *et al.*, 1997). Their ability to model problems of arbitrary complexity makes them a promising tool for the description of potential energy functions in molecules (Tafeit *et al.*, 1996; Kyoung *et al.*, 1997). Also, neural nets are attractive tools for the description of torsional potentials for internal rotation and conformational

* To whom correspondence should be addressed.

problems. In this context, the usual approach is to develop a Fourier expansion of the potential energy on the angular coordinates. However, when several coordinates must be considered, the complexity of the expansion increases exponentially with the number of coordinates. Here, neural networks are suitable tools to describe the multidimensional hypersurface and to handle it numerically. Thus, neural networks can be applied to the numerical description of multidimensional potential hypersurfaces for dynamical studies. Another application is the numerical integration of the potential terms appearing in the variational treatment of large amplitude vibrations. In this field, the interest of neural networks is mainly descriptive rather than predictive, in contrast with QSAR studies.

An example of the study of conformational problems by neural networks, is the description of the torsional potential for cofactor (6R, 1'R,2'S)-5,6,7,8-tetrahydrobiopterin (Tafeit *et al.*, 1996). In that work however, the full range of torsional data was not well reproduced, the periodicity found was wrong, and the global error was very high. No explanation of these results was found, and the study focussed on the non-periodic zone around the global minimum. In the present work, we analyse this problem and the usefulness of neural networks to describe torsional periodic potentials for large amplitude vibrations in its full range of definition and in non-periodic zones.

2. THEORY

In the initial works on the ability of neural nets to reproduce a function to arbitrary precision, it was assumed that a non decreasing squashing activation function needs to be used (Hecht-Nielsen, 1987; Hecht-Nielsen, 1989; Hornik, *et al.*, 1989). Since then, the most used activation function has been the sigmoidal one,

$$g_S(x) = \frac{1}{1 + \exp(-x - \alpha)} \tag{1}$$

where α, the bias, is a threshold parameter. The sigmoidal function is continuous and differentiable, properties of interest for the application of learning algorithms. This function is, by definition, a squashing function, since it performs a mapping of the real line,

$$g_S : \Re \rightarrow (0, 1). \tag{2}$$

Equation (1) shows that the function saturates quickly for large or small values of x.

In modeling problems, to approximate a function, $f(x_1, x_2, \ldots, x_m)$, the net must be trained using a set of several known patterns of input values $\{(x_1, x_2, \ldots, x_m)\}$ and the corresponding output values of the function. The first step in this process is a random initialization of weights and bias in a given interval, usually (−1,1). Then, the training procedure modifies the weights of the links between units and the bias to minimize the overall error between the desired output and the output computed by the net. The full set of patterns is presented iteratively and the global error is minimized step by step. Each of these steps involving the full set of patterns is called an "epoch". When trained, the net can be used in place of the function, from a numerical point of view. In contrast to Taylor or Fourier series, the application of neural nets to the description of functions does not need to give beforehand any functional form to the functional approach.

Backpropagation is currently the most widely applied neural network architecture and the one applied in the previous studies of potential energy hypersurfaces (Tafeit *et al.*, 1996; Kyoung *et al.*, 1997). The information processing it carries out is the approximation of a mapping or function from an m-dimensional to an n-dimensional Euclidean space, f: $\Re^m \rightarrow \Re^n$ (Hecht-Nielsen, 1989). Learning in this model is attained using the backpropagation algorithm (Hecht-Nielsen, 1989; Rojas, 1996), which works as follows. A neural network is an implementation of a composite function, the network function, from the input to the output space. The network function is defined through weights (w) and bias (α), and the learning problem is the determination of the appropriate values of these parameters. Different learning methods have been devised, generally based in algorithms for the search of a minimum, and among them backpropagation. Here, we define an error function, E,

$$E(\mathbf{W}) = (1/2)\sum_{i=1}^{p}(o_i - t_i)^2 \tag{3}$$

where $\mathbf{W}$ is a column vector collecting weights and bias, p represents the number of training patterns, o is the network output and t is the training output. Expanding E in a Taylor series on $\mathbf{W}$ in the vicinity of a point, $\mathbf{W}_0$, we get,

$$E(\mathbf{W}) = E(\mathbf{W}_0) + [\nabla E(\mathbf{W}_0)]^{\mathrm{T}}(\mathbf{W}_0 - \mathbf{W}_1) + (1/2)(\mathbf{W}_0 - \mathbf{W}_1)^{\mathrm{T}}\mathbf{H}(\mathbf{W}_0 - \mathbf{W}_1) + \ldots \tag{4}$$

where $\mathbf{H}$ is the hessian of the error function. Differentiation of equation (4) leads to

$$\nabla E(\mathbf{W}_1) \simeq \nabla E(\mathbf{W}_0) + \mathbf{H} \cdot (\mathbf{W}_0 - \mathbf{W}_1). \tag{5}$$

Assuming the new point is a minimum, equation (5) yields,

$$\mathbf{W}_1 = \nabla E(\mathbf{W}_0) - \mathbf{H}(\mathbf{W}_0)^{-1} \cdot \mathbf{W}_0. \tag{6}$$

Equation (6) expresses the Newton method for multidimensional minimization. In the simplest case the hessian is taken as a constant diagonal matrix with equal diagonal elements. Thus, we arrive at

$$\mathbf{W}_1 = \nabla E(\mathbf{W}_0) - \eta \cdot \mathbf{W}_0 \tag{7}$$

which is the steepest descent method. Equation (7) is the basis of the backprogation algorithm. Backpropagation uses steepest descent to minimize the error between the observed and the computed outputs. The specific details of the updating algorithm for weights and bias can be found elsewhere (Hecht-Nielsen, 1989; Rojas, 1996). In this context, the η parameter is known as the learning rate. The learning rate represents the size of the step taken in

the descent direction. Thus, a large value corresponds to a large motion on the error hypersurface, whereas a small step produces small motions on the hypersurface useful to locate the closest minimum.

A problem that complicates the use of neural networks in the modelling of functions is overfitting. Overfitting appears when the network learns the training set correctly, even perfectly, but interpolates unknown points erroneously. This behaviour appears because the network function adapts to describe all the training data, but it is very different from the actual function. A way to identify this problem is to use a validation set of data. This validation set contains several data not collected in the training set. By comparison of the errors in the training and validation sets overfitting can be identified. To some extent, overfitting is related to the number of parameters to consider in the training process.

3. RESULTS AND DISCUSSION

Our aim is to study the ability of neural networks to describe periodic potentials, both in its full range of definition and in local zones. To perform the study we will generate the different data sets using a known potential function. Thus, we select the potential for the internal rotation of the methyl group in acetaldehyde (Muñoz-Caro *et al.*, 1995),

$$V(\theta) = 197.26 - 204.24\cos(3\theta) + 6.98\cos(6\theta) \quad (8)$$

where θ is the torsional angle.

We start with the description of the potential in its full range of definition. Thus, a training set is build using the data of θ (torsional coordinate) and the potential from 0° to 360° in increments of 10°. All the potential data are scaled in the interval [0,1] dividing by the highest barrier height, 408.48 cm^{-1}. To control overfitting, a validation set is constructed using values of θ and the potential from 5° to 355°, also in increments of 10°.

Our first approach is the construction of a fully connected backpropagation network using the standard sigmoidal activation function. This is the approach previously used in the modeling of potential hypersurfaces (Tafeit *et al.*, 1996; Kyoung *et al.*, 1997). In this case, we need to scale appropriately the input data. Thus, the torsional angle is divided by 2π. A one-dimensional function can be modelized by a neural network using only one hidden layer and the sigmoidal activation function (Cybenko, 1989; Müller *et al.*, 1995). Therefore, we use the usual three-layered network. The optimal number of parameters (weights and bias) to use cannot be deduced beforehand. However, in QSAR studies by neural networks (So and Karplus, 1996) the best results are found with an approximate patterns to parameters ratio of 2:1. Although these results cannot be rigorously generalized, they give us a good starting point. Thus, for our 36 training patterns we will work with a number of units in the hidden layer ranging from three to seven. Some initial tests have shown that a learning rate of 0.9 with 60,000 epochs gives the best results. The results obtained are collected in Table 1. In all cases, the error in the validation set is very close to the error for the training set. Therefore, no problems of overfitting seem to appear. We observed a drop of the average square error for six units, i.e. 19 parameters. A significant observation is that different initializations of the networks lead, sometimes, to different final errors. This fact is also observed when initialization intervals different from the usual (−1,1) are used.

Table 1. Neural modeling of the potential for the torsion of the methyl group of acetaldehyde in its full range of definition. N is the number of units in the hidden layer, ASE is the average square error (quotient between the sum of squares error and the number of patterns), and M is the number of parameters in the training process. The activation function is the sigmoidal function. Learning rate and number of training epochs: 0.9 and 60000, respectively. The subscripts t and v refer to the training and validation sets

N	M	ASE_t	ASE_v
3	10	0.0619	0.0610
4	13	0.0450	0.0424
5	16	0.0442	0.0412
6	19	0.0052	0.0066
7	22	0.0050	0.0060

Usually, increasing the complexity of the network improves the results, but the risk of overfitting increases as well. A way to keep the number of parameters under control is to use more appropriate activation functions. Kolmogorov's theorem ensures the proper operation of a neural network using nonlinear activation functions, but it does not determine which are the best activation functions for a given problem. However, it is possible to define a closeness index, I, in a similar way to a quantum mechanical overlap integral. Thus,

$$I = \int_{\theta_1} \cdots \int_{\theta_n} f(\theta_1, \ldots, \theta_n) g(\theta_1) \ldots g(\theta_n) d\tau \quad (9)$$

where f is the function modeled by the neural net and the gs are activation functions. When f is a periodic function, and taking into account that it can be expanded in a Fourier series, we have,

$$\int_{\theta_1} \cdots \int_{\theta_n} f(\theta_1, \ldots, \theta_n) g_s(\theta_1) \ldots g_s(\theta_n) d\tau < \int_{\theta_1} \cdots \int_{\theta_n} f(\theta_1, \ldots, \theta_n) g_p(\theta_1) \ldots g_p(\theta_n) d\tau \quad (10)$$

where g_s is the sigmoidal activation function and g_p is a periodic activation function. From equation (10), we can expect the error obtained with a periodic activation function to be smaller than the error obtained with the sigmoidal function. As an appropriate periodic function, we define in this work a generalized cosine activation function,

$$g_p(\theta; \alpha) = \cos(\theta + \alpha) \quad (11)$$

which, in a sense, is a variant of the Gallant and White squashing function (Gallant and White, 1988). This new function is a periodic squashing function that performs a map of the real line,

$$g_p : \Re \rightarrow [-1, 1]. \quad (12)$$

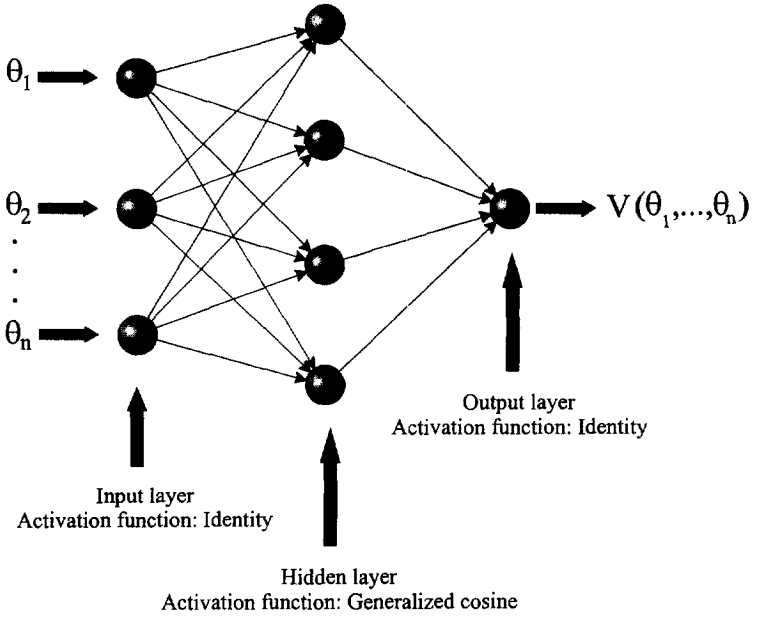

Fig. 1. Structure of a feedforward, fully connected backpropagation neural network for simulating Fourier series.

The α parameter is a phase factor (bias) that increases the flexibility of the function. This parameter is included in the training of the network. The function is a valid activation function since arbitrary nonlinearity is sufficient to represent all functions by neural networks (Kreinovich, 1991). We have implemented the generalized cosine activation function in the Kernel of the Stutgart Neural Network Simulator, SNNS 4.1 (Zell *et al.*, 1995).

Using the new activation function, we build "Fourier " networks as shown in Fig. 1 (in the input and output layers the activation function is the identity). Thus, the input values can be introduced directly in radians. The output values (potential) are scaled in the interval [0,1] as exposed previously. In these neural networks, the weight of the links between the input and hidden layers and the bias define the periodicity of the net.

We build a backpropagation network containing one hidden layer with three units to mimic the constant, the $\cos(3\theta)$ and the $\cos(6\theta)$ terms of the original potential, equation (8). Thus, the number of units in the input–hidden–output layers is 1:3:1. Initial tests have shown that the best results are obtained with only 4000 epochs and learning rates of 0.2 for the 2000 first epochs and 0.1 for the 2000 last epochs. In the training, the average square error reaches a minimum of 0.0001. We observe that the error in the validation set follows closely the variation of the error in the training set during the learning, reaching a final value of 0.0001. Then, as in the previous case, no overfitting seems to arise. As expected, the error found, with only three units, is much smaller than with the sigmoidal activation function using more processing units. However, as in the sigmoidal case, we found that different initializations can lead to different final errors.

In the present case, the close relationship between the activation function and the modeled function permits analysis of this problem examining the values of weights and bias. Figure 2 shows three cases. In case (a), we use a Fourier network with the standard random initialization in the interval (−1,1). The result shows one link from the hidden to the output layer with zero weight. This net corresponds to a pure $\cos(3\theta)$ expansion. In case (b), we initialize weights and bias in the interval (−6, 6) to introduce higher periodicity. The result shown in Fig. 2, is obtained frequently in different training runs. Here, a 39-fold solution is found with an average square error for the training set of 0.0001. In contrast, the error for the validation set reaches 0.5000. This overffiting problem appears because the network function increases its periodicity to fit the training data. The local nature of this behaviour is clear in case (c). Here, we build a net that mimics the original Fourier expansion, equation (8), initializing the weights to 0.0, 3.0 and 6.0 in the input–hidden links and to 0.5, 0.5 and 0.1 in the hidden–output links. All bias are set to 0.0. In this case, the net converges to the physically correct solution, i.e. Equation (8), with a average square error for the training and validation sets of 0.0000.

To investigate the modeling of a non-periodic zone of a periodic potential, we have taken the surroundings of a maximum in the potential function, equation (8). Thus, we generate a training set using values of θ ranging from 30° to 90° in increments of 5°. A corresponding validation set is constructed using values of θ from 31° to 86° , also in increments of 5°. In all cases, the output is scaled, dividing by the barrier. Using the sigmoidal activation function, models with three and four processing units are tested with the input scaled by 2π. The results obtained after 60,000 training epochs with

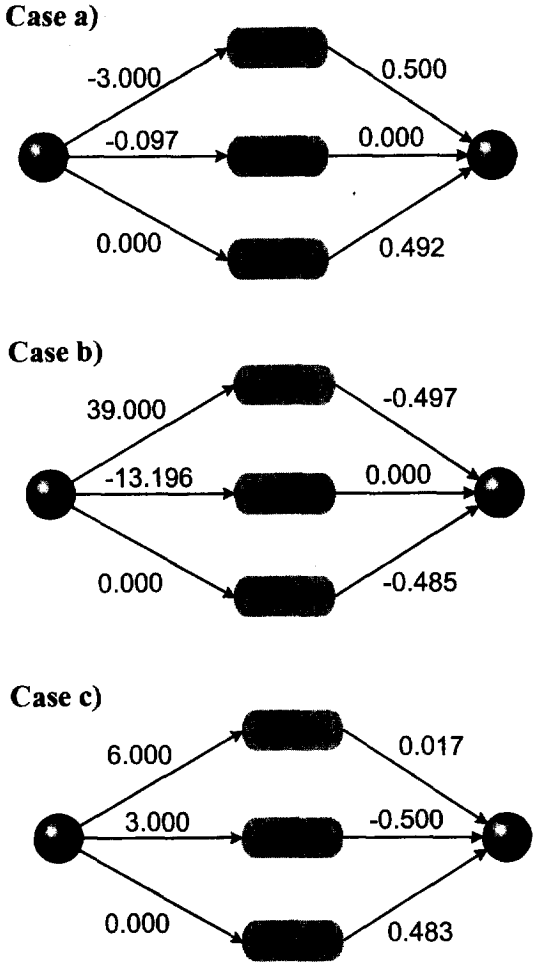

Fig. 2. Fourier networks for the torsional potential of the methyl group of acetaldehyde in its full range of definition. The weights are represented on the links and the bias in the shadowed boxes.

learning rate of 0.9 are shown in Table 2. Now, the minimum error is found 0.0075, whereas the validation set exhibits an error very close to this value, 0.00616. The present average square error is smaller than the one found in the study of the full interval of definition for the same number of units, see Table 1. We also test a Fourier network with three and four processing units in the hidden layer. The results after 4000 epochs with learning rates of 0.2, in the first 2000, and 0.1 in the next 2000, are also collected in Table 2. In all cases, the validation set error is similar to the training error. We find that for a given number of processing units, the error is much smaller than the corresponding error obtained with the sigmoidal activation function. The minimum error found was 0.0001. Thus, the generalized cosine activation function produces better results than the sigmoidal activation function consuming fewer computational resources.

The observed behaviour of our neural models can be analysed in the following form. In a neural net, the output, *out*, is a composite, nonlinear function of weights, w, and bias, α. Thus, with N_i, N_h and one units, in the input, hidden and output functions, respectively, we have,

$$out = g^0\left(\sum_{j=1}^{N_h} w_j^0 g_j^h\left(\sum_{k=1}^{N_i} w_k^i g_k^i(x_k + \alpha_k^i) + \alpha_j^h\right) + \alpha^o\right) \tag{13}$$

where g denotes activation function and the superscripts i, h and o refer to the input, hidden and output layers, respectively. Equation (13) shows that in the output function, weights and bias appear as arguments. Since activation functions are nonlinear, the output depends nonlinearly on w and α. From equation (13), the global error, E, in the training process is,

$$E = (1/2)\sum_{l=1}^{p}\left\{g^o\left(\sum_{j=1}^{N_h} w_j^o g_j^h\left(\sum_{k=1}^{N_i} w_k^i g_k^i(x_k + \alpha_k^i) + \alpha_j^h\right) + \alpha^o\right) - y_l\right\}^2 \tag{14}$$

where p is the number of test cases and y_l represents the desired output. Equation (14) shows that E depends nonlinearly on w and α. In consequence, the derivatives of E respect to weights and bias also depend on w and α. Since the initial arguments, i.e. w and α, are selected at random, we have the possibility of starting our network function, and its derivatives, with different periodicities. Even assuming that the network can modelize the desired output, the training process by itself is unable to find the optimal value of arguments except if we start close to them. This fact can be proved, using the backpropagation algorithm, in the following form. For two steps in the training process, equation (7) gives the actualization mechanism, which is a simplification of Newton's method, equation (6). The condition for the Newton direction, $-\mathbf{H}(\mathbf{W})^{-1}\,\nabla E(\mathbf{W})$, to be a descent direction, i.e. that the error decreases respect to all the components of $\mathbf{W}$, is (Dennis and Schnabel, 1996),

$$-[\nabla \mathbf{E}(\mathbf{W}_i)]^{\mathrm{T}} \cdot \mathbf{H}(\mathbf{W}_i)^{-1} \cdot \nabla \mathbf{E}(\mathbf{W}_i) < 0 \tag{15}$$

which is true for a positive definite hessian, $\mathbf{H}(\mathbf{W})$. In backpropagation, this condition reduces to a positive learning rate, η, see equation (7). A positive definite hessian means that we are in, or close to, a

Table 2. Neural modeling of the potential for the methyl torsion of acetaldehyde in the interval $\theta = 30°$–$90°$. N represents the number of processing units in the hidden layer, M the number of parameters (weights and bias) and ASE is the average square error (quotient between the sum of squares error and the number of patterns). The subscripts t and v refer to the training and validation sets, respectively. Case (a) sigmoidal activation function. Case (b) generalized cosine activation function

N	M^a	ASE_t^a	ASE_v^a	M^b	ASE_t^b	ASE_v^b
3	10	0.0373	0.0306	9	0.0001	0.0001
4	13	0.0075	0.0066	12	0.0002	0.0002

minimum. In addition, since a descent direction always decreases the function respect to all the coordinates it is not possible to overcome barriers. The result is that the training will converge to the minimum closest to the starting point. Since the initial position on the error hypersurface depends on the initial, random, values of weights and bias, we can converge to different minima for different starting values of the parameters. This explains why we can obtain different results in different initializations of our networks. The problem cannot be overcome using training techniques based in Newton's method (backpropagation, backpropagation with momentum, Rprop, Quickprop, Qrprop or scaled conjugate gradient, among others) since all of them perform local minimizations. On the other hand, global strategies such as genetic algorithms can find, in principle, the global minimum. However, the network function is only an approach to the actual function. Thus, it is not possible to identify the global minimum with the correct, most physically significant, one.

This problem depends, to some extend, on the patterns included in the training set. For instance, if only maxima and minima for an n periodical function are included, any network function with periodicity $k \cdot n$, with k integer, will reproduce exactly the training set. Thus, we need to use a training set obtained from a representative sample of the potential. However, this is a necessary but not sufficient condition, since we cannot assign beforehand any given periodicity to the network function.

On the other hand, when the function to modelize is non-periodic, or a non-periodic zone of a periodic function, the situation differs. Periodicity is not a problem here, and different starting arguments will lead to the same minimum in the error function. Also, since periodicity does not need to be reproduced, the actual function can easily be mimicked with non-periodic composite functions. This fact explains why using the sigmoidal function the error in the non-periodic case is smaller than in the periodic one.

These results can be generalized to more than one dimension because they only depend on the nature of the actual function and of the learning algorithm. In this case, however, the use of the generalized cosine presents an advantage over the sigmoidal activation function. A network with two hidden layers using sigmoidal activation functions suffices for the representation of arbitrary functions of any number of continuous arguments (Funahashi, 1989; Müller et al, 1995). However, any periodic function can be approximated with our periodic activation function using only one hidden layer. This can be proved in the following form. Let $f(\theta_1, \theta_2, \ldots, \theta_n)$ be a periodic function of n variables. Expanding it in a Fourier series we have,

$$f(\theta_1, \theta_2, \ldots, \theta_n) = \prod_{i=1}^{n} \sum_{j=0}^{\infty} [a_{ij} \cos(j \cdot \theta_i) + b_{ij} \sin(j \cdot \theta_i)]. \tag{16}$$

Taking into account that $\sin(\theta) = \cos(\theta - \pi/2)$ and rearranging,

$$f(\theta_1, \theta_2, \ldots, \theta_n) = \sum_{j=1}^{\infty} c_j \prod_{i=1}^{n} \cos(d_j \cdot \theta_i + \beta_j) \tag{17}$$

where the β are phase parameters. We will show that equation (17) can be implemented using a three layered Fourier network.

Let us consider a three layered neural network using the generalized cosine activation function with n input units, one for each variable, N_h units in the hidden layer and one output unit. In this net, the network output is obtained as,

$$\text{out} = \sum_{k=1}^{N_h} w_k^h \cos\left(\sum_{j=1}^{n} [w_j^i \theta_j + \alpha_j]\right) \tag{18}$$

where the superscripts h and i refer to hidden and input layers, respectively. Taking into account that,

$$\cos\left(\sum_{j=1}^{n} [a_j \theta_j]\right) = \cos(a_1\theta_1)\cos\left(\sum_{j=2}^{n} [a_j\theta_j]\right) - \cos(a_1\theta_1 - \pi/2)\cos\left(\sum_{j=2}^{n} [a_j\theta_j] - \pi/2\right) \tag{19}$$

we obtain,

$$\cos\left(\sum_{j=1}^{n} a_j\theta_j\right) = \sum_{j=1}^{2^{n-1}} \prod_{l}^{n} \cos(a_l\theta_l + \gamma_j) \tag{20}$$

where the γ are parameters. Thus, applying equation (20) to equation (18) we obtain,

$$out = \sum_{k=1}^{N_h} w_k^h \sum_{j=1}^{2^{n-1}} \prod_{l}^{n} \cos(w_l^i\theta_l + \beta_j) \tag{21}$$

which is equivalent to equation (17).

The ability of the present approach to deal with multidimensional cases can be illustrated with the two-dimensional methyl torsional problem of acetone. The potential for this problem as a function of the two torsional angles, θ_1 and θ_2 , is (Smeyers *et al.*, 1993):

$$\begin{aligned} V(\theta_1, \theta_2) =& 332.34 - 200.76(\cos(3\theta_1) + \cos(3\theta_2)) \\ &+ 64.87\cos(3\theta_1)\cos(3\theta_2) - 0.16(\cos(6\theta_1) \\ &+ \cos(6\theta_2)) + 2.35(\cos(6\theta_1)\cos(3\theta_2) \\ &+ \cos(3\theta_1)\cos(6\theta_2)) \\ &- 0.08\cos(6\theta_1)\cos(6\theta_2) \\ &- 80.08\sin(3\theta_1)\sin(3\theta_2). \end{aligned} \tag{22}$$

We will model one of the maxima of the function. Thus, a training set is generated by performing a grid on the θ_1 and θ_2 angles from 0° to 120° in increments of 20°. The validation set is obtained from a grid on θ_1 and θ_2 from 5° to 105°, also in increments of 20°. The energy data are scaled dividing each value by the barrier, 793.62 cm^{-1}. A three layered Fourier network is constructed with three units (nine parameters) in the hidden layer. After

2000 epochs with learning rate of 0.2 and 2000 epochs with learning rate of 0.1, we obtain average square errors of 0.0029 and 0.0024 for the training and validation sets, respectively. Thus, our Fourier neural network reproduces the two-dimensional potential function with only one hidden layer.

4. CONCLUSIONS

We have analysed the use of backpropagation neural networks for describing periodic potential hypersurfaces. We have found that neural networks using the standard sigmoidal activation function do not reproduce easily the input data when modeling a periodic function in its full range of definition. Also, we found that different initializations can lead to different values of the error, even with wrong periodicity. This fact explains the bad results obtained by (Tafeit *et al.*, 1996) when representing a full torsional hypersurface.

To reduce the number of parameters in the training process, we define a periodic activation function. Using this function, we find errors much smaller than those obtained using sigmoidal activation functions. Moreover, the number of training epochs is dramatically reduced. However, it is still possible to find different solutions as a function of the starting parameters. We show that this problem is a consequence of the existence of several not equivalent minima on the error hypersurface. This means that neural networks can be used to modelize periodic functions obtaining small errors, but they do not, necessarily, describe properly the periodicity of the problem.

On the other hand, description of local, non-periodic zones of a periodic function using the usual activation functions or periodic activation functions, reproduces the actual function. In addition, the periodic activation function reproduces much better the data using less parameters and training epochs than the sigmoidal activation function.

Finally, we found that the new activation function can modelize a periodic function of any number of arguments using only one hidden layer.

REFERENCES

Bulsari, A. and Saxén, H. (1991) *Neurocomputing* **3**, 125.

Cybenko, G. (1989) *Math. Control Signals Systems* **2**, 303.

Dennis, J. E. Jr and Schnabel, R. B. (1996) *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. SIAM, Philadelphia.

Fisher, T. H., Petersen, W. P. and Lüthi, H. P. (1995) *J. Comput. Chem.* **16**, 923.

Funahashi, K-I. (1989) *Neural Networks* **2**, 183.

Gallant, A. R. and White, H. (1988) *IEEE Second International Conference on Neural Networks* **Vol I**, 657.

Hecht-Nielsen, R. (1987) *IEEE First International Conference On Neural Networks* **Vol. III**, 11.

Hecht-Nielsen, R. (1989) *Proceedings of The International Joint Conference on Neural Networks* **Vol. I**, 593.

Hopfield, J. J. (1982) *Proc. Natl. Acad. Sci. USA* **79**, 2554.

Hornik, K., Stinchconbe, M. and White, H. (1989) *Neural Networks* **2**, 359.

Ivanciuc, O., Ravine, J. P., Cabrol-Bass, D., Panaye, A. and Doucet, J. P. (1997) *J. Chem. Inf. Comput. Sci.* **37**, 587.

Kireev, D. B. (1995) *J. Chem. Inf. Comput. Sci.* **35**, 175.

Kreinovich, V. Y. (1991) *Neural Networks* **4**, 381.

Kyoung, T. N., Byung, H. C., Su, Y. K., Mu, S. J. and Scheraga, H. A. (1997) *Chem. Phys. Lett.* **271**, 152.

McCulloch, W. S. and Pitts, W. (1943) *Bull. Math. Biophys.* **5**, 115.

Minsky, M. and Papert, S. (1969) *Perceptrons: an Introduction to Computational Geometry*. MIT Press, Cambridge, MA.

Müller, B., Reinhardt, J. and Strickland, M. T. (1995) *Neural Networks*. An introduction. Springer, Berlin.

Muñoz-Caro, C., Niño, A. and Moule, D. C. (1995) *J. Chem. Soc. Faraday Trans.* **91**, 399.

Peterson, K. L. (1990) *Phys. Rev. A* **41**, 2457.

Rojas, R. (1996) *Neural Networks*. Chapt. 7. pp. 149–182. Springer, Berlin. .

Smeyers, Y. G., Senent, M. L., Botella, V. and Moule, D. C. (1993) *J. Chem. Phys.* **98**, 2754.

So, S-S. and Karplus, M. (1996) *J. Med. Chem.* **39**, 5246.

Sprecher, D. A. (1965) *Transactions of The American Mathematical Society* **115**, 340.

Tafeit, E., Estelberger, W., Horejsi, R., Moeller, R., Oettl, K., Vrecko, K. and Reibnegger, G. (1996) *J. Mol. Graphics* **14**, 12.

Zell, A., Mamier, G., Vogt, M., Mache, N., Hübner R., Döring, S., Herrmann, K.-U., Soyez, T., Schmalzl, M., Sommer, T., Hatzigeorgiou, A., Posselt, D., Schreiner, T., Kett, B., Clemente, G. and Wieland, J. External contributions by: Reczko, M., Riedmiller, M., Seemann, M., Ritt, M., DeCoster, J., Biedermann, J., Danz, J., Wehrfritz, C., Werner, R., Berthold, M. and Orsier, B. (1995) *SNNS (Stuttgart Neural Network Simulator)*, Version 4. 1.

Zupan, J. and Gasteiger, G. (1993) *Neural Networks for Chemists*. VCH, New York.