Journal of Molecular Structure (Theochem), 277 (1992) 87–107
Elsevier Science Publishers B.V., Amsterdam

Application of neural networks with feedback connections in chemistry: prediction of ^{13}C NMR chemical shifts in a series of monosubstituted benzenes

V. Kvasnička, Š. Sklenák and J. Pospíchal

Department of Mathematics, Faculty of Chemical Technology, Slovak Technical University, CS-81237 Bratislava (Czech and Slovak Federal Rep.)

(Received 27 February 1992)

Abstract

A simple steepest-descent adaptation process of neural networks with feedback connections (i.e. with oriented cycles) is described. The method is illustrated with an application of this type of neural network to the prediction of ^{13}C NMR chemical shifts in a series of monosubstituted benzenes.

INTRODUCTION

A paradigm of neural networks [1–4] has come to play an important role in chemistry as a computing tool for the classification of chemical objects and the prediction of their properties. An excellent review of applications and the meaning of neural networks in chemistry and related fields of research has recently been published by Zupan and Gasteiger [3], in which an extensive list of references can be found. Almost all applications of neural networks in chemistry are performed by the so-called feed-forward networks with a back-propagation method [4] for the adaptation process. Graph-theoretically, these networks may be considered as connected oriented graphs without oriented cycles. The condition of acyclicity of networks is very important for the simple recurrent evaluation of neuron activities and partial derivatives of the objective function to be minimized. Removing this condition of acyclicity leads to serious numerical problems. In particular, the neuron activities cannot be evaluated recurrently: they

Correspondence to: V. Kvasnička, Department of Mathematics, Faculty of Chemical Technology, Slovak Technical University, CS-81237 Bratislava, Czech and Slovak Federal Rep.

0166-1280/92/$05.00 © 1992 Elsevier Science Publishers B.V. All rights reserved.

have to be determined by a string of coupled non-linear equations that may be solved only iteratively (providing they have a common solution — a fixed point). Fortunately, the partial derivatives are still determined by a system of linear equations. The main difference between their evaluation in acyclic and cyclic networks is that in the former case these equations are in a form allowing their recurrent solution, whereas in the latter the equations can be solved by a standard method of linear algebra.

The problem of inclusion feedback connections to layered neural networks has been studied by many authors [5–7]. All these approaches are usually based either on a consideration of neural networks as dynamic systems determined by a system of coupled differential equations with relatively complicated structure and interpretation, or on the definition of neural networks of different classes that are attached to the original one. The purpose of this paper is to demonstrate a simple method of adapting neural networks to the possibility that they may contain, in general, oriented cycles. It is shown that this problem may be solved in a way closely related to the standard adaptation process used in feed-forward (i.e. acyclic) neural networks — the steepest-descent adaptation process. The method is illustrated with two examples. The first example corresponds to a standard problem: how to use neural networks as a classifier of 0,1-vectors for symmetry. The second example might be of interest for chemical applications of neural networks: the neural networks with cycles are used as a predictor of ^{13}C NMR chemical shifts of a series of monosubstituted benzenes.

BASIC CONCEPTS

Here, we present only the basic concept of neural networks — the details may be found in the literature [1–4]. Formally, a neural network is determined [8–9] as an oriented connected graph $\mathbb{G} = (V,E)$ (see Fig. 1) where $V = \{v_1, v_2, \ldots, v_N\}$ is a non-empty set composed of N vertices — neurons. A set $E = \{e_1, e_2, \ldots, e_M\}$ is composed of M edges — connections. Each connection $e \in E$ is interpreted as an ordered pair of neurons from V; $e = [v, v']$. We say that this connection is outgoing from the vertex v and incoming to the vertex v′. The set V of neurons is divided into three disjoint subsets (see Fig. 1)

$$V = V_I \cup V_H \cup V_O \tag{1}$$

where V_I is composed of N_I input neurons that are incident only with outgoing connections, V_H is composed of N_H hidden neurons that are incident at least with one incoming and one outgoing connection, and V_O is composed of N_O output neurons that are incident at least with one incoming connection. In our forthcoming considerations we shall always

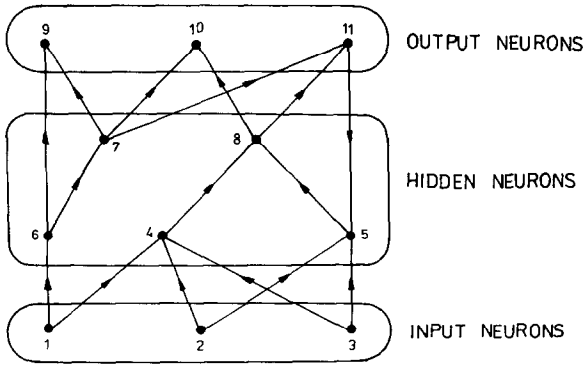

Fig. 1. An example of a neural network determined as a connected and oriented graph. This neural network is composed of eleven neurons: neurons labelled 1–3 are input neurons; 4–8 are hidden neurons; 9–11 are output neurons; it also contains an oriented cycle composed of neurons labelled 5, 8 and 11.

assume that the subsets V_I and V_O are non-empty, i.e. neural networks must contain at least one input and one output neuron.

Neurons and connections of neural networks are evaluated by real numbers (see Fig. 2). We assign to each hidden and/or output neuron v_i a threshold coefficient ϑ_i and to each connection $e = [v_i, v_j]$ a weight coefficient ω_{ji}. Moreover, we assign to each neuron v_i an activity x_i. We postulate that activities of input neurons are constant whereas the activities of other neurons are determined by

$$x_i = f(\xi_i) \qquad (i = 1, 2, \ldots, N) \tag{2a}$$

$$\xi_i = \sum_j \omega_{ij} x_j + \vartheta_i \tag{2b}$$

where the summation index j runs over all neurons that are adjacent with the neuron v_i by connections that are incoming to v_i and outgoing from v_j.

The transfer function $f(\xi)$ is a positive monotonically increasing function that fulfills the asymptotic conditions $f(\xi) \to B$ as $\xi \to \infty$ and $f(\xi) \to A$ as $\xi \to -\infty$, where $A < B$. For instance, these requirements are simply met if the transfer function is

$$f(\xi) = \frac{B + A\exp(-\xi)}{1 + \exp(-\xi)} \tag{3a}$$

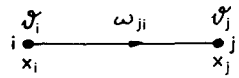

Fig. 2. A connection $e = [v_i, v_j]$ is evaluated by the weight coefficient ω_{ji}, and its vertices v_i and v_j are simultaneously evaluated by threshold coefficients ϑ_i and ϑ_j and activities x_i and x_j.

with the first derivative determined by

$$f'(\xi) = \frac{[-A + f(\xi)][B - f(\xi)]}{B - A} \tag{3b}$$

This function maps the set $\mathbb{R}$ of all real numbers onto an open interval (A,B); f: $\mathbb{R} \rightarrow (A,B)$. Most frequently, the transfer function is applied either for $A = 0$, $B = 1$ or for $A = -1$, $B = 1$. The first case corresponds to a classical sigmoidal function, which maps the set $\mathbb{R}$ onto an open interval (0,1), whereas the second choice corresponds to an analog of a hyperbolic tangent function mapping $\mathbb{R}$ onto the open interval $(-1,1)$.

For neural networks without oriented cycles it is easy to show [10] that neurons may be indexed in such a way that vertex v_i is incident with the incoming (outgoing) connections with initial (terminal) neurons indexed by $j < i$ ($j > i$). This means that input (output) neurons should be indexed by the lowest (greatest) indices of $\{1, 2, \ldots, N\}$, whereas hidden neurons are indexed by intermediate indices that are greater (smaller) than those used for the indexing of input (output) neurons. According to this simple graph-theoretical property of acyclic neural networks, the system of equations (eqns. (2a) and (2b)) for activities may be solved recurrently; an activity x_i is then determined by the previous activities $x_1, x_2, \ldots, x_{i-1}$. First we calculate all activities of neurons that are juxtaposed to the input neurons (their activities are kept fixed during the whole calculation). In the next step we calculate the activities of neurons that are juxtaposed to the neurons of activities calculated in the previous step. This process is recurrently repeated until all activities are calculated.

Unfortunately, the above recurrent method for calculation of activities cannot be applied to neural networks containing cycles. Here, eqns. (2a) and (2b) represent a string of coupled non-linear equations and the activities of hidden and output neurons are obtained only by their iterative solution. This is the main obstacle to a broader application of neural networks with cycles: their activities are not determined in such an easy way as the activities of neural networks without cycles.

All activities form a state vector $x = (x_1, x_2, \ldots, x_N)$. This vector is divided into three subvectors composed of input, hidden and output activities

$$x = x_I \oplus x_H \oplus x_O \tag{4}$$

The neural network with fixed weight and threshold coefficients may be formally considered as a function

$$F: \mathbb{R}^{N_I} \rightarrow (A,B)^{N_O} \tag{5}$$

which assigns to an input vector x_I (descriptor) an output vector x_O (classifier) composed of entries from the open interval (0,1):

$$F(x_I) = x_O \tag{6}$$

The hidden activities are not displayed explicitly here; they only play the role of intermediate results.

An adaptation process of neural network involves looking for such threshold and weight coefficients that for a pair of the prescribed input and output vectors x_I and $\hat{x}_O$ give an output vector x_O, determined by eqn. (6), "closely" related to the prescribed $\hat{x}_O$. Let us construct an objective function

$$E = \tfrac{1}{2}(x_O - \hat{x}_O)^2 = \tfrac{1}{2}\sum_k g_k^2 \tag{7a}$$

$$g_j = \begin{cases} (x_j - \hat{x}_j) & (\text{for } j \in V_O) \\ 0 & (\text{for } j \notin V_O) \end{cases} \tag{7b}$$

where x_k and $\hat{x}_k$ are entries of x_O and $\hat{x}_O$ respectively. A goal of an adaptation process is to find the weight and threshold coefficients that will minimize the objective function E.

This minimization may be carried out by a version of the gradient method [11], e.g. by its simplest version called the steepest-descent method [11]. Therefore we have to know all partial derivatives $\partial E/\partial\omega_{ji}$ and $\partial E/\partial\vartheta_j$ of the objective function with respect to weight and threshold coefficients respectively. These partial derivatives may be expressed as

$$\frac{\partial E}{\partial \omega_{ji}} = \frac{\partial E}{\partial x_j}\frac{\partial x_j}{\partial \omega_{ji}} = \frac{\partial E}{\partial x_j} f'(\xi_j) x_i \tag{8a}$$

and

$$\frac{\partial E}{\partial \vartheta_j} = \frac{\partial E}{\partial x_j}\frac{\partial x_j}{\partial \vartheta_j} = \frac{\partial E}{\partial x_j} f'(\xi_j) \tag{8b}$$

Comparing these two equations we arrive at a very important relationship between derivatives $\partial E/\partial\omega_{ji}$ and $\partial E/\partial\vartheta_j$:

$$\frac{\partial E}{\partial \omega_{ji}} = \frac{\partial E}{\partial \vartheta_j} x_i \tag{9}$$

This means that the whole process of calculation of partial derivatives $\partial E/\partial\omega_{ji}$ and $\partial E/\partial\vartheta_j$ may be reduced to a substantially simpler calculation of $\partial E/\partial\vartheta_j$; the two-index derivatives $\partial E/\partial\omega_{ji}$ are then determined by one-index derivatives $\partial E/\partial\vartheta_j$ and corresponding activities x_i. The partial derivative $\partial E/\partial x_j$ from the right-hand side of eqn. (8b) is expressed as

$$\frac{\partial E}{\partial x_j} = \frac{\partial E}{\partial x_j}\frac{\partial x_j}{\partial x_j} + \sum_l \frac{\partial E}{\partial x_l}\frac{\partial x_l}{\partial x_j} = g_j + \sum_l \frac{\partial E}{\partial x_l} f'(\xi_l)\omega_{lj} = g_j + \sum_l \frac{\partial E}{\partial \vartheta_l}\omega_{lj} \tag{10}$$

where in the third expression the relation shown in eqn. (8b) has been used. Introducing this result in eqn. (8b) we arrive at the final equation for partial

derivatives $\partial E/\partial \vartheta_j$:

$$\frac{\partial E}{\partial \vartheta_j} = f'(\xi_j)\left(g_j + \sum_l \frac{\partial E}{\partial \vartheta_l}\,\omega_{lj}\right) \text{ (for } j \in V_H \cup V_O) \tag{11}$$

where the term g_j is determined by eqn. (7b) and the summation runs over all neurons that are adjacent to the neuron v_j by incoming connections.

For acyclic neural networks, eqn. (11) immediately provides the well-known recurrent relations used in the so-called back-propagation adaptation [4]. The partial derivatives $\partial E/\partial \vartheta_j$ (for $j \in V_O$) are simply determined by $\partial E/\partial \vartheta_j = f'(\xi_j)g_j$. These derivatives are then used for the evaluation of derivatives $\partial E/\partial \vartheta_j$, where the index j corresponds to neurons adjacent to output neurons by outgoing connections. The same process is recurrently repeated until all derivatives $\partial E/\partial \vartheta_j$ assigned to hidden neurons are calculated. Finally, knowing all partial derivatives $\partial E/\partial \vartheta_j$, the derivatives $\partial E/\partial \omega_{ji}$ are simply determined by eqn. (9). It is quite surprising that the same relation (eqn. (11)) is satisfied also for a neural network with cycles, although now it could not be applied recurrently. The partial derivatives $\partial E/\partial \vartheta_j$ are determined by eqn. (11) as a system of linear equations. Its form allows their determination iteratively; successive application of the results from the left-hand side to its right-hand side gives the partial derivatives with the prescribed precision after a finite number of steps.

The method of calculation of partial derivatives outlined above may be simply generalized for more than one pair of input–output vectors x_I and $\hat{x}_O$

$$x_I^{(1)}/\hat{x}_O^{(1)},\ x_I^{(2)}/\hat{x}_O^{(2)},\ \ldots,\ x_I^{(r)}/\hat{x}_O^{(r)} \tag{12}$$

which form the so-called training set. The objective function is then determined by

$$E = \sum_{i=1}^{r} E^{(i)} \tag{13a}$$

$$E^{(i)} = \tfrac{1}{2}(x_O^{(i)} - \hat{x}_O^{(i)})^2 \tag{13b}$$

where $x_O^{(i)}$ is the output vector of neural networks determined by eqn. (6) as a response to an input vector $x_I^{(i)}$, and $\hat{x}_O^{(i)}$ are required output vectors assigned to input vectors $x_I^{(i)}$. Partial derivatives of the generalized objective function are then equal to the sum of the partial derivatives of $E^{(i)}$ evaluated by means of eqns. (9) and (11).

If we knew the gradient of the objective function, then the adaptation process of a neural network could be realized by a minimization of the objective function with respect to the threshold and weight coefficients, so that output activities $x_O^{(1)}, x_O^{(2)}, \ldots, x_O^{(r)}$ would be closely related to the required output activities $\hat{x}_O^{(1)}, \hat{x}_O^{(2)}, \ldots, \hat{x}_O^{(r)}$, which are assigned to input activities $x_I^{(1)}, x_I^{(2)}, \ldots, x_I^{(r)}$. The steepest-descent minimization method is

based on the following updating of the threshold and weight coefficients:

$$\omega_{ji}^{(k+1)} = \omega_{ji}^{(k)} - \lambda \frac{\partial E}{\partial \omega_{ji}} \tag{14a}$$

$$\vartheta_j^{(k+1)} = \vartheta_j^{(k)} - \lambda \frac{\partial E}{\partial \vartheta_j} \tag{14b}$$

where the positive parameter $\lambda > 0$ should be sufficiently small to ensure convergence of the adaptation process and simultaneously sufficiently large to achieve fast convergence. We have experienced very good computational efficiency [8,9] if the adaptation process of neural networks is carried out by more sophisticated versions of the gradient method [11], e.g. by the method of conjugate gradients or by the method of variable metric.

Unfortunately the above comment is correct only for a neural network without cycles. If the network contains cycles, the adaptation process is a much more numerically complex problem because the activities and the derivatives of objective functions are not determined recurrently but via strings of coupled equations. In particular, the adaptation process is now composed of two parts: first, the weight and threshold coefficients are updated such that the objective function is monotonically decreasing; secondly, each change of these weight and threshold coefficients involves also the necessity of updating the activities of hidden and output neurons in order to keep activities self-consistent. Both these problems should be solved simultaneously; in the $(k+1)$th iterative step of the adaptation process, activities, partial derivatives, weight and threshold coefficients are recurrently updated on the basis of their values from the previous kth step

$$x_i^{(k+1)} = f(\xi_i^{(k)}) \tag{15a}$$

$$\xi_i^{(k)} = \sum_j \omega_{ij}^{(k)} x_j^{(k)} + \vartheta_i^{(k)} \tag{15b}$$

$$\left(\frac{\partial E}{\partial \vartheta_j}\right)^{(k+1)} = f'(\xi_j^{(k)}) \left[g_j^{(k)} + \sum_l \left(\frac{\partial E}{\partial \vartheta_l}\right)^{(k)} \omega_{lj}^{(k)} \right] \tag{15c}$$

$$g_j^{(k)} = \begin{cases} (x_j^{(k)} - \hat{x}_j) & (\text{for } j \in V_O) \\ 0 & (\text{for } j \notin V_O) \end{cases} \tag{15d}$$

$$\left(\frac{\partial E}{\partial \omega_{ji}}\right)^{(k+1)} = \left(\frac{\partial E}{\partial \vartheta_j}\right)^{(k+1)} x_i^{(k)} \tag{15e}$$

$$\omega_{ji}^{(k+1)} = \omega_{ji}^{(k)} - \lambda \left(\frac{\partial E}{\partial \omega_{ji}}\right)^{(k+1)} \tag{15f}$$

$$\vartheta_j^{(k+1)} = \vartheta_j^{(k)} - \lambda \left(\frac{\partial E}{\partial \vartheta_j}\right)^{(k+1)} \tag{15g}$$

The last two equations (eqns. (15f) and (15g)) express the fact that the neural network is adapted by the steepest-descent method (see eqns. (14a) and (14b)). The coefficient λ in eqns. (15f) and (15g) should be so small that for each step of the iterative solution the updated values of partial derivatives and activities belong to tight neighborhoods of their exact values (for the current values of weight and threshold coefficients). Equations (15a) and (15b) represent a simple iterative solution of activities. Similarly, eqns. (15c)–(15e) are a transcription of eqn. (11) in a recurrent form applicable to the iterative determination of partial derivatives of the objective function. A generalization of these equations for the objective function determined by eqns. (13a) and (13b), which corresponds to a training set composed of r objects (eqn. (12)), is straightforward. In the framework of each iterative step, eqns. (15a)–(15e) are applied separately to each object from the training set; then the partial derivatives $(\partial E/\partial \omega_{ji})^{(k)}$ and $(\partial E/\partial \vartheta_j)^{(k)}$ are determined as the sum of partial derivatives assigned to single objects.

The objective function E is highly non-linear, so its minimization is a non-standard numerical task. Usually, a local minimum is achieved in cases where the value of the objective function is much greater than a value in the global minimum. It means that it is necessary to perform at least a few adaptation processes with randomly generated initial values of threshold and weight coefficients. Then we select those coefficients that give the lowest (positive) value of the minimized objective functions for the forthcoming active process of the neural network in which input activities are determined by descriptors of objects taken from the so-called testing set.

The extrapolation outside the training set is another critical point of applications of neural networks as a classifier of objects from the testing set. It may very often happen that the active process gives results of classification with much lower precision than required. Then the adaptation process of the neural network should be repeated with new randomly generated threshold and weight coefficients. New coefficients obtained as a result of this adaptation process are again used for the forthcoming active process. If the classification of objects thus produced in the active process does not satisfy the required precision repeatedly, then it is necessary to turn the user's attention to a topology of the neural network or to descriptors (input activities) of objects from training and testing sets: the topology of the neural network is very probably inadequate for the problem under study, or the descriptors do not properly reflect the internal structure of the objects.

AN ILLUSTRATIVE EXAMPLE

The correctness of the present theory of neural networks with cycles

TABLE 1

Training set of three-dimensional 0,1-vectors composed of eight pairs of input and output pair activity vectors

Input activity							
$x_I^{(1)}$	$x_I^{(2)}$	$x_I^{(3)}$	$x_I^{(4)}$	$x_I^{(5)}$	$x_I^{(6)}$	$x_I^{(7)}$	$x_I^{(8)}$
0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
0	1	0	1	0	1	0	1
Required output activity							
$\hat{x}_O^{(1)}$	$\hat{x}_O^{(2)}$	$\hat{x}_O^{(3)}$	$\hat{x}_O^{(4)}$	$\hat{x}_O^{(5)}$	$\hat{x}_O^{(6)}$	$\hat{x}_O^{(7)}$	$\hat{x}_O^{(8)}$
0.9	−0.9	0.9	−0.9	−0.9	0.9	−0.9	0.9

outlined in the previous section is illustrated and tested in an attempt to classify three-dimensional 0,1-vectors $(x_1,x_2,x_3) \in \{0,1\}^3$. If for a vector (x_1,x_2,x_3) its first and third components are equal (i.e. $x_1 = x_3$), then we call this vector symmetric (evaluated as 0.9), and in the opposite case (i.e. $x_1 \neq x_3$) it is called asymmetric (evaluated as −0.9). A training set composed of all eight possible three-dimensional 0,1-vectors is displayed in Table 1. Two different neural networks (see Fig. 3) are used for the classification of vectors for symmetry. The constants A and B from the transfer function (eqn. (3)) are determined as $A = -1$ and $B = 1$. These networks are only different because one of them (labelled (B) in Fig. 3) contains a simple

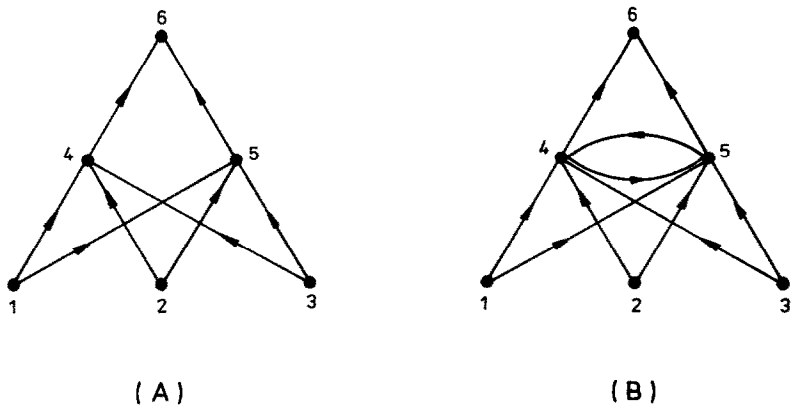

Fig. 3. Two different neural networks composed of three input neurons, two hidden neurons, and one output neuron. These networks are used for the classification of three-dimensional 0,1-vectors. The network labelled (B) contains a simple oriented cycle composed of neurons labelled 4 and 5.

TABLE 2

Results of neural network (A) in Fig. 3

Hidden activity							
$x_H^{(1)}$	$x_H^{(2)}$	$x_H^{(3)}$	$x_H^{(4)}$	$x_H^{(5)}$	$x_H^{(6)}$	$x_H^{(7)}$	$x_H^{(8)}$
0.76	1.00	0.77	1.00	−0.77	0.77	−0.77	0.77
0.76	−0.77	0.77	−0.77	1.00	0.77	1.00	0.77
Output activity							
$x_O^{(1)}$	$x_O^{(2)}$	$x_O^{(3)}$	$x_O^{(4)}$	$x_O^{(5)}$	$x_O^{(6)}$	$x_O^{(7)}$	$x_O^{(8)}$
0.89	−0.89	0.90	−0.89	−0.89	0.90	−0.89	0.90
Weight coefficient[a]							
$\omega_{41} = -4.07\ (-1)$		$\omega_{42} = 0.02\ (1)$		$\omega_{43} = 4.09\ (1)$		$\omega_{51} = 4.09\ (1)$	
$\omega_{52} = 0.02\ (1)$		$\omega_{53} = -4.07\ (-1)$		$\omega_{64} = 4.40\ (1)$		$\omega_{65} = 4.40\ (1)$	
Threshold coefficient[a]							
$\vartheta_4 = 2.01\ (1)$		$\vartheta_5 = 2.01\ (1)$		$\vartheta_6 = -3.85\ (1)$			
$E = 0.15 \times 10^{-3}$		$\lvert\text{grad } E\rvert = 0.23 \times 10^{-2}$					

[a]Numbers in parentheses denote the starting values of the coefficients.

cycle composed of neurons 4 and 5, whereas the other network (labelled (A) in Fig. 3) does not contain any cycle. The adaptation process for both neural networks has been performed by the steepest-descent method discussed in the previous section; the starting values of weight and threshold coefficients are listed in Tables 2 and 3 (numbers placed in parentheses). The resulting values of hidden activities and the corresponding adapted values of weight and threshold coefficients are listed in Tables 2 and 3. We see that for both neural networks the classification of vectors achieved is the same as the required classification. The adaptation process used was accelerated by the so-called momentum method; the updating equations (eqns. (15f) and (15g)) were changed by

$$\omega_{ji}^{(k+1)} = \omega_{ji}^{(k)} - \lambda\left(\frac{\partial E}{\partial \omega_{ji}}\right)^{(k+1)} + \alpha\Delta\omega_{ji}^{(k)} \tag{16a}$$

$$\vartheta_j^{(k+1)} = \vartheta_j^{(k)} - \lambda\left(\frac{\partial E}{\partial \vartheta_j}\right)^{(k+1)} + \alpha\Delta\vartheta_j^{(k)} \tag{16b}$$

where the terms $\Delta\omega_{ji}^{(k)}$ and $\Delta\vartheta_j^{(k)}$ are the changes in weight and threshold coefficients, respectively, in the previous adaptation step. The parameter α usually ranges between 0.5 and 0.9. The starting value of the steepest-

TABLE 3

Results of neural network (B) in Fig. 3

Hidden activity							
$x_H^{(1)}$	$x_H^{(2)}$	$x_H^{(3)}$	$x_H^{(4)}$	$x_H^{(5)}$	$x_H^{(6)}$	$x_H^{(7)}$	$x_H^{(8)}$
0.66	1.00	0.67	1.00	−0.85	0.67	−0.84	0.68
0.66	−0.85	0.67	−0.84	1.00	0.67	1.00	0.68
Output activity							
$x_O^{(1)}$	$x_O^{(2)}$	$x_O^{(3)}$	$x_O^{(4)}$	$x_O^{(5)}$	$x_O^{(6)}$	$x_O^{(7)}$	$x_O^{(8)}$
0.89	−0.89	0.89	−0.89	−0.89	0.89	−0.89	0.90
Weight coefficient[a]							
$\omega_{41} = -3.37\ (-1)$		$\omega_{42} = 0.05\ (1)$		$\omega_{43} = 3.42\ (1)$		$\omega_{45} = -2.16\ (-1)$	
$\omega_{51} = 3.42\ (1)$		$\omega_{52} = 0.05\ (1)$		$\omega_{53} = -3.37\ (-1)$		$\omega_{54} = -2.16\ (-1)$	
		$\omega_{64} = 4.83\ (1)$		$\omega_{66} = 4.83\ (1)$			
Threshold coefficient[a]							
$\vartheta_4 = 3.03\ (1)$		$\vartheta_5 = 3.03\ (1)$		$\vartheta_6 = -3.60\ (1)$			
$E = 0.31 \times 10^{-3}$		$\lvert\text{grad } E\rvert = 0.31 \times 10^{-2}$					

[a]Numbers in parentheses denote the starting values of the coefficients.

descent parameter λ is unity. If we observe that the value of objective function is increasing, then the parameter λ is modified by $\lambda \to \beta\lambda$, where $0 < \beta < 1$. Unfortunately, application of a momentum method needs approximately twice as much computer memory, but for computers currently in use and equipped with 5–10 MB memory, this fact does not involve additional complications.

The adaptation process of our two neural networks, with initial values of weight and threshold coefficients listed in Tables 2 and 3, was finished after approximately 5000 iterations. Such a huge number of iterations that are necessary for the adaptation process may involve some speculation about the numerical efficiency of the suggested method. Recently, we have used [8,9], for the adaptation process of a neural network without cycles, much more powerful optimization methods, in particular the method of conjugated gradients or the method of variable metric. Both these approaches reduce the number of iterations, usually to a few hundred. One of their basic ideas is that the parameter λ (the length of a direction vector) should be optimized so that the objective function achieves a minimum value in the given direction. The efficiency obtained by the reduction in the number of

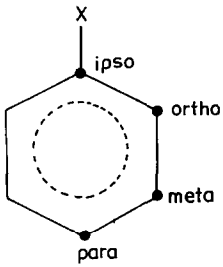

Fig. 4. A monosubstituted benzene; –X represents a substituent; four different positions on the benzene skeleton (ipso, ortho, meta and para) are distinguished.

iterations is then spoiled by the minimization of λ, which corresponds to a few dozen additional optimizations that are almost as numerically demanding as the simple steepest-descent method. The neural networks with cycles could not be adapted by a version of modern gradient methods; simultaneous optimization of weight coefficients and activities in these neural networks requires the performance of the adaptation process by the simplest gradient method, i.e. the steepest-descent method. The application here of either the method of conjugated gradients or the method of variable metric may cause very dramatic changes in weight coefficients. Then the activities updated using eqns. (12a) and (12b) are not closely related to their exact values for current values of weight coefficients, i.e. the principal requirement of our adaptation process is not fulfilled.

APPLICATION TO A PREDICTION OF ^{13}C NMR CHEMICAL SHIFTS IN A SERIES OF MONOSUBSTITUTED BENZENES

The present theory of neural networks with cycles is applied as a classifier of monosubstituted benzenes with respect to their ^{13}C NMR chemical shifts (four different positions on the benzene skeleton; see Fig. 4). The neural networks used are composed of 11 input neurons (their activities are equal to descriptors, which determine in a proper way the structure of a given substitutent –X) and four output neurons (with activities equal to chemical shifts in ipso, ortho, meta and para positions respectively; see Fig. 5). The chemical shifts of monosubstituted benzenes belong to a relatively large interval of real numbers. Therefore they should be compressed (see ref. 8) to values from a smaller open interval (A,B) by the following non-linear transformation:

$$f(x) = \frac{B + Ae^{-\alpha(x+\beta)}}{1 + e^{-\alpha(x+\beta)}} \tag{17}$$

The coefficients α and β are adjusted so that $x_{\min} \leqslant x \leqslant x_{\max}$ is mapped onto $A + \varepsilon \leqslant y \leqslant B - \varepsilon$. A positive number ε is determined by $\varepsilon = (B - A)\kappa$ for a

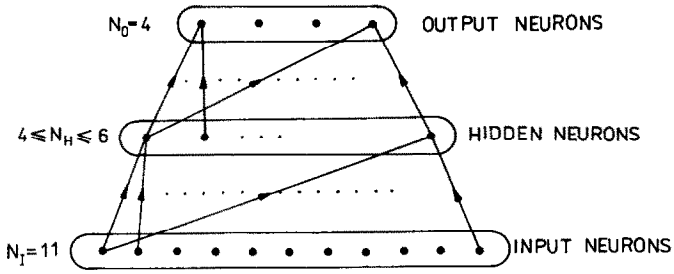

Fig. 5. Schematic plot of neural networks used for the classification of monosubstituted benzenes. These networks are composed of 11 input neurons, four output neurons, and between four and six hidden neurons. The hidden neurons induce a complete oriented subgraph, i.e. each hidden neuron is adjacent to each other hidden neuron by two connections of opposite direction.

small positive κ in the range $0 < \kappa < 1/2$. An inverse of eqn. (17) is

$$x = f^{-1}(y) = \frac{1}{\alpha}\ln\frac{y - A}{A + y} - \beta \tag{18}$$

The coefficients α and β are determined by

$$\beta = -\tfrac{1}{2}(x_{\min} + x_{\max}) \tag{19a}$$

and

$$\alpha = \frac{1}{x_{\min} + \beta}\ln\frac{\varepsilon}{B - A - \varepsilon} \tag{19b}$$

The hidden neurons of the used neural networks (see Fig. 5) ($4 \leqslant N_H \leqslant 6$) form a complete subgraph, i.e. each hidden neuron is adjacent to each other hidden neuron by two oriented connections of opposite direction.

The 11 descriptors (x_1–x_{11}) assigned to a substituent –X are formed by the same method as in our recent publication [9] devoted to the similar problem of predicting the yields of nitration in the meta position of monosubstituted benzenes. For the present purpose the first-level descriptors do not contain entries corresponding to a charge; therefore the substituent groups –X are now always without a charge. This means that the first-level descriptors are composed of three entries (x_1–x_3). The second-level descriptors remain unchanged and are composed of four original entries (x_4–x_7). Finally, the same descriptors as in the second level are used for the third-level descriptors (x_8–x_{11}). Illustrative examples are given in Fig. 6 and Table 4 (for the interpretation of single entries see ref. 9). The results are listed in Table 5 with experimental values of chemical shifts [12]. The training (testing) set is composed of 44 (20) monosubstituted benzenes that are determined (in particular, their substituent groups –X) by the 11 non-negative integers specified above. The best results are achieved for neural networks

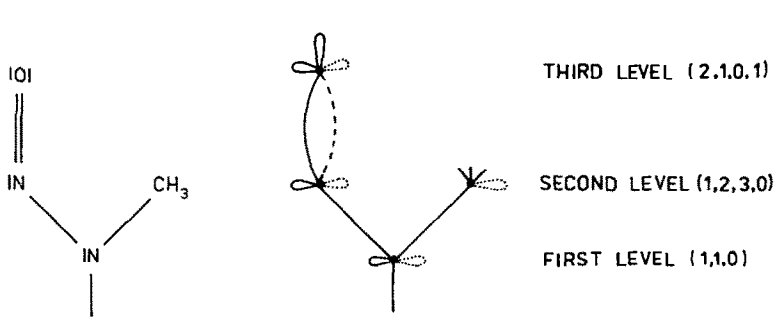

Fig. 6. An illustrative example of the construction of eleven descriptors (input activities) of the substituent $-N(CH_3)NO$ (for details see ref. 9).

composed of six hidden neurons (denoted NN6 in Table 5). In the ipso position, chemical shifts tend to have greater values than in the three remaining positions, so the adaptation process results in trained neural networks, which are able to predict these shifts in satisfactory agreement with the corresponding experimental values. Unfortunately, the results obtained for the ortho, meta and para positions are not as good as those for the ipso position. This is because the values of their chemical shifts fluctuate near zero; hence the predictions of these chemical shifts represent a difficult task for solving by neural networks. A possible way of surmounting this problem is to apply the neural network approach separately to each position. Then the problem of great differences in chemical shifts for different positions does not arise.

DISCUSSION

It seems that a generalization of feed-forward neural networks in such a way that they may potentially contain feedback connections might be of value not only as a theoretical achievement but also as a proper and flexible classifier of objects of chemical interest. Unfortunately, the second conjecture has not been confirmed by our present calculations. In particular, a detailed analysis of neural networks with cycles has failed to give a satis-

TABLE 4

Ilustrative examples of descriptors assigned to simple substitutents –X[a]

–X	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}
$-CH_3$	0	1	3	0	0	0	0	0	0	0	0
$-NO_2$	0	1	0	4	2	0	2	0	0	0	0
$-N(CH_3)NO$	1	1	0	1	2	3	0	2	1	0	1

[a]For details see ref. 9.

TABLE 5

Experimental and predicted ^{13}C NMR chemical shifts of monosubstituted benzenes R–Ph

No.	–R		Chemical shift			
			Ipso	Ortho	Meta	Para
Training set						
1	–H	Exp	0.0	0.0	0.0	0.0
		NN4	−0.6	−1.6	0.3	−0.6
		NN5	0.3	0.8	−0.1	−1.1
		NN6	0.3	0.8	0.3	−0.1
2	$-CH_3$	Exp	9.2	0.7	−0.1	−3.1
		NN4	10.1	0.3	0.3	−2.1
		NN5	9.0	1.2	−0.1	−1.6
		NN6	9.4	−0.1	0.3	−2.6
3	$-CH_2CH_3$	Exp	15.6	−0.5	0.0	−2.7
		NN4	15.9	−3.1	0.3	−3.1
		NN5	15.9	−1.6	−0.1	−2.6
		NN6	15.1	−0.6	−0.1	−2.1
4	$-CH(CH_3)_2$	Exp	20.1	−2.0	0.0	−2.5
		NN4	18.1	−4.9	0.3	−4.3
		NN5	18.6	−2.6	−0.1	−4.3
		NN6	18.6	−1.6	−0.1	−3.1
5	$CH_2CH=CH_2$	Exp	15.3	0.0	0.2	−2.4
		NN4	13.5	−2.6	0.3	−3.1
		NN5	14.7	−1.6	−0.1	−3.1
		NN6	14.3	−0.6	−0.1	−2.6
6	$-CH_2C\equiv N$	Exp	1.7	0.5	−0.8	−0.7
		NN4	2.1	0.8	0.3	−0.1
		NN5	2.1	1.7	0.3	−2.1
		NN6	2.1	1.2	−0.1	0.3
7	$-CH_2COOH$	Exp	4.2	0.4	1.2	−0.9
		NN4	4.5	1.7	0.3	−1.1
		NN5	4.1	1.2	0.3	−0.1
		NN6	4.1	1.7	0.3	−0.6
8	$-CH_2Si(CH_3)_3$	Exp	12.0	−0.1	0.0	−4.2
		NN4	11.6	−1.1	0.3	−3.1
		NN5	12.7	0.8	−0.1	−2.6
		NN6	11.6	−0.1	−0.1	−3.1
9	$-CH_2NH_2$	Exp	14.9	1.4	−0.1	−1.9
		NN4	13.1	−1.1	−0.1	−1.1
		NN5	13.5	−1.1	−0.1	−2.1
		NN6	15.1	−1.1	−0.1	−2.1

TABLE 5 (continued)

No.	–R		Chemical shift			
			Ipso	Ortho	Meta	Para
10	$-CH_2NO_2$	Exp	2.2	2.2	2.2	1.2
		NN4	1.1	1.7	0.3	−0.1
		NN5	2.5	1.7	0.8	1.2
		NN6	1.7	0.8	0.3	0.8
11	$-CH_2OH$	Exp	12.4	−1.2	0.2	−1.1
		NN4	11.6	−0.1	−0.1	−0.1
		NN5	11.6	−1.1	0.3	−1.6
		NN6	13.9	−1.1	−0.1	−1.1
12	$-CH_2OCH_3$	Exp	11.0	0.5	−0.4	−0.5
		NN4	11.2	−0.1	−0.1	0.3
		NN5	10.8	−0.1	−0.1	−1.1
		NN6	10.8	0.8	−0.1	−1.6
13	$-CH_2SCH_3$	Exp	9.8	0.4	−0.1	−1.6
		NN4	11.2	−0.1	−0.1	0.3
		NN5	10.5	1.2	−0.1	−0.6
		NN6	11.2	0.8	−0.1	−1.6
14	$-CH_2F$	Exp	8.5	−0.7	0.4	0.5
		NN4	10.5	0.3	−0.1	0.8
		NN5	8.3	−0.1	−0.1	0.3
		NN6	8.3	0.3	−0.1	0.3
15	$-CF_3$	Exp	2.5	−3.2	0.3	3.3
		NN4	3.7	1.7	0.3	5.3
		NN5	2.5	−2.6	−0.1	2.9
		NN6	2.9	−1.6	−0.1	2.9
16	$-CH_2Cl$	Exp	9.3	0.3	0.2	0.0
		NN4	10.5	0.3	−0.1	0.8
		NN5	10.1	0.3	0.3	0.3
		NN6	9.0	0.8	−0.1	−0.1
17	$-CH=CH_2$	Exp	8.9	−2.3	−0.1	−0.8
		NN4	9.0	−0.6	0.3	−3.1
		NN5	9.0	−1.6	0.3	−2.6
		NN6	9.0	−2.6	0.3	−1.6
18	$-C\equiv CH$	Exp	−6.2	3.6	−0.4	−0.3
		NN4	−6.9	4.1	0.8	0.8
		NN5	−8.4	2.9	0.8	−0.1
		NN6	−10.2	3.7	0.3	1.2
19	$-C\equiv N$	Exp	−15.7	3.6	0.7	4.3
		NN4	−8.4	5.3	0.8	4.5
		NN5	−8.4	4.9	0.8	4.5
		NN6	−10.2	3.7	0.8	2.9

TABLE 5 (continued)

No.	–R		Chemical shift			
			Ipso	Ortho	Meta	Para
20	–CHO	Exp	8.4	1.2	0.5	5.7
		NN4	6.8	0.8	0.3	2.1
		NN5	9.0	1.2	0.8	5.3
		NN6	8.3	2.1	0.8	5.3
21	$-CONH_2$	Exp	5.0	−1.2	0.1	3.4
		NN4	4.9	1.2	0.3	4.5
		NN5	6.4	0.8	0.3	4.5
		NN6	4.9	−1.1	0.3	3.3
22	–COOH	Exp	2.1	1.6	−0.1	5.2
		NN4	3.7	1.2	0.3	5.7
		NN5	2.9	2.1	−0.1	6.0
		NN6	2.9	0.8	0.3	5.7
23	–COF	Exp	4.3	1.6	−0.7	5.3
		NN4	4.1	0.8	0.3	6.8
		NN5	3.3	−0.1	−0.1	5.3
		NN6	4.1	1.2	0.3	6.8
24	$-SiH_3$	Exp	−0.1	8.0	0.3	2.0
		NN4	−1.6	7.2	−0.1	1.2
		NN5	−2.1	6.8	−0.1	0.8
		NN6	−0.1	7.9	−0.1	1.2
25	–COCl	Exp	4.7	2.7	0.3	6.6
		NN4	4.1	0.8	0.3	6.8
		NN5	4.1	2.1	0.3	6.8
		NN6	4.1	1.7	0.3	6.4
26	$-NH_2$	Exp	18.2	−13.4	0.8	−10.0
		NN4	18.1	−5.5	0.3	−7.6
		NN5	19.1	−7.6	0.3	−5.5
		NN6	18.1	−11.1	0.8	−6.9
27	$-NHCH_3$	Exp	21.4	−16.2	0.8	−11.6
		NN4	24.9	−10.2	0.3	−8.4
		NN5	26.4	−11.1	0.3	−7.6
		NN6	22.9	−13.4	0.8	−7.6
28	$-N(CH_3)_2$	Exp	22.5	−15.4	0.9	−11.5
		NN4	27.3	−11.1	0.3	−8.4
		NN5	27.3	−11.1	0.3	−8.4
		NN6	25.6	−12.2	0.3	−7.6
29	$-NHCOCH_3$	Exp	9.7	−8.1	0.2	−4.4
		NN4	11.2	−4.3	0.3	−4.9
		NN5	10.8	−5.5	0.8	−4.3
		NN6	10.5	−6.2	0.8	−4.9

TABLE 5 (continued)

No.	-R		Chemical shift			
			Ipso	Ortho	Meta	Para
30	$-NHNH_2$	Exp	22.8	−16.5	0.5	−9.6
		NN4	24.2	−9.3	0.3	−6.2
		NN5	24.9	−11.1	0.8	−6.9
		NN6	23.5	−13.4	0.8	−7.6
31	$-N(CH_3)NO$	Exp	23.7	−9.5	0.8	−1.4
		NN4	22.9	−10.2	0.8	−4.3
		NN5	24.2	−10.2	1.2	−1.6
		NN6	20.6	−13.4	0.8	−5.5
32	$-NO_2$	Exp	19.9	−4.9	0.9	6.1
		NN4	19.6	−4.9	0.8	5.7
		NN5	18.1	−6.2	1.2	4.9
		NN6	19.1	−5.5	1.2	4.9
33	-NCS	Exp	3.0	−2.7	1.3	−1.0
		NN4	3.3	−2.1	1.2	0.3
		NN5	2.1	−4.9	1.2	−0.6
		NN6	2.9	−1.6	0.8	−1.1
34	-OH	Exp	26.9	−12.8	1.4	−7.4
		NN4	28.2	−11.1	0.8	−6.9
		NN5	30.2	−16.2	1.2	−7.6
		NN6	28.2	−12.2	1.2	−6.2
35	$-OCOCH_3$	Exp	22.4	−7.1	0.4	−3.2
		NN4	21.1	−10.2	1.2	−2.1
		NN5	20.1	−11.1	1.2	−3.7
		NN6	21.7	−11.1	1.2	−5.5
36	$-OSi(CH_3)_3$	Exp	26.8	−8.4	0.9	−7.1
		NN4	25.6	−10.2	0.8	−5.5
		NN5	24.9	−10.2	0.3	−9.3
		NN6	27.3	−9.3	0.8	−5.5
37	-SH	Exp	2.1	0.7	0.3	−3.2
		NN4	1.7	−0.1	0.8	−3.7
		NN5	2.5	−0.6	0.3	−2.6
		NN6	2.1	0.8	0.8	−1.6
38	-SCN	Exp	−3.7	2.5	2.2	2.2
		NN4	−4.9	0.3	0.8	1.1
		NN5	−3.7	2.9	0.8	2.5
		NN6	−4.3	1.2	0.8	1.7
39	$-SOCH_3$	Exp	17.6	−5.9	1.1	2.4
		NN4	17.2	−5.5	0.8	0.3
		NN5	15.5	−3.7	0.8	−0.1
		NN6	17.7	−6.9	1.2	1.7

TABLE 5 (continued)

No.	–R		Chemical shift			
			Ipso	Ortho	Meta	Para
40	$-SO_2CH_3$	Exp	12.3	−1.4	0.8	5.1
		NN4	12.3	−1.1	0.3	3.3
		NN5	13.9	−0.6	0.8	4.9
		NN6	12.3	−1.1	0.8	6.0
41	$-SO_2Cl$	Exp	15.6	−1.7	1.2	6.8
		NN4	15.1	−3.1	0.8	6.0
		NN5	15.5	−2.1	1.2	7.5
		NN6	16.8	−1.1	0.8	7.5
42	–F	Exp	34.8	−13.0	1.6	−4.4
		NN4	36.0	−14.7	1.7	−4.3
		NN5	31.4	−16.2	1.2	−4.9
		NN6	32.7	−11.1	1.2	−4.9
43	$-C(CH_3)_3$	Exp	22.1	−3.4	−0.4	−3.1
		NN4	19.1	−5.5	0.3	−4.9
		NN5	19.6	−3.7	−0.1	−5.5
		NN6	22.3	−3.7	−0.1	−4.9
44	$-CH_2C(CH_3)_3$	Exp	10.6	1.5	−1.0	−3.1
		NN4	10.1	−0.1	0.3	−3.1
		NN5	10.5	1.2	−0.1	−2.6
		NN6	10.5	0.3	−0.1	−3.1
Testing set						
45	$-CH_2COCH_3$	Exp	6.0	1.0	0.2	−1.6
		NN4	7.9	−0.1	0.3	−2.1
		NN5	1.2	3.3	0.3	−1.1
		NN6	5.7	2.5	−0.1	−1.1
46	$-CH_2N(CH_3)_2$	Exp	11.1	0.8	−0.2	−1.5
		NN4	12.3	−0.6	−0.1	−1.1
		NN5	11.2	0.8	−0.1	−1.6
		NN6	11.6	0.8	−0.1	−2.1
47	$-CH_2SOCH_3$	Exp	0.8	1.5	0.4	−0.2
		NN4	10.1	−0.1	−0.1	−0.6
		NN5	10.8	0.3	0.3	−0.1
		NN6	8.6	1.2	−0.1	−1.6
48	$-CH_2SO_2CH_3$	Exp	−0.1	2.1	0.6	0.6
		NN4	6.4	−0.1	0.3	−1.6
		NN5	2.5	1.7	0.3	−1.1
		NN6	5.7	2.1	−0.1	−1.1
49	$-CCl_3$	Exp	16.3	−1.7	−0.1	1.8
		NN4	3.7	1.7	0.3	4.9
		NN5	−6.2	6.4	−0.6	7.2
		NN6	5.3	−2.1	−0.1	1.2

TABLE 5 (continued)

No.	–R		Chemical shift			
			Ipso	Ortho	Meta	Para
50	$-CH_2Br$	Exp	9.5	0.7	0.3	0.2
		NN4	10.5	0.3	−0.1	0.8
		NN5	10.1	1.2	0.3	0.8
		NN6	9.4	0.8	−0.1	−0.6
51	$-CH_2I$	Exp	10.5	0.0	0.0	−0.9
		NN4	10.5	0.3	−0.1	0.8
		NN5	9.7	2.1	0.3	0.8
		NN6	9.7	0.8	−0.1	−1.1
52	$-C(CH_3)=CH_2$	Exp	12.6	−3.1	−0.4	−1.2
		NN4	18.1	−6.2	0.3	−4.9
		NN5	19.1	−4.9	0.3	−4.3
		NN6	14.3	−5.5	0.3	−2.1
53	$-COCH_3$	Exp	8.9	0.1	−0.1	4.4
		NN4	7.9	−0.1	0.3	2.1
		NN5	13.5	−3.1	0.3	0.3
		NN6	11.2	−4.3	0.3	0.8
54	$-COCH_2CH_3$	Exp	8.8	0.2	−0.5	4.3
		NN4	5.3	0.3	0.3	2.5
		NN5	13.9	−3.7	0.3	−1.1
		NN6	10.5	0.3	0.3	−0.6
55	$-CON(CH_3)_2$	Exp	8.0	−1.5	−0.2	1.0
		NN4	0.6	2.9	0.3	5.3
		NN5	9.7	−1.1	0.3	0.8
		NN6	12.7	3.7	0.3	1.2
56	$-COOCH_3$	Exp	2.0	1.2	−0.1	4.3
		NN4	0.8	2.1	0.3	6.0
		NN5	2.5	2.9	−0.1	6.0
		NN6	9.7	2.5	0.3	2.5
57	$-Si(CH_3)_3$	Exp	11.7	5.9	−0.7	0.3
		NN4	12.7	−2.1	−0.1	−2.1
		NN5	10.1	2.1	−0.1	−0.6
		NN6	12.3	2.1	−0.1	0.3
58	$-NHCH_2CH_3$	Exp	20.0	−15.7	0.7	−11.4
		NN4	23.5	−9.3	0.3	−8.4
		NN5	23.5	−8.4	0.3	−6.9
		NN6	24.9	−11.1	0.3	−8.4
59	$-N(CH_2CH_3)_2$	Exp	19.3	−16.5	0.6	−13.0
		NN4	24.9	−10.2	0.3	−8.4
		NN5	22.9	−6.9	−0.1	−6.9
		NN6	28.2	−8.4	−0.1	−7.6

TABLE 5 (continued)

No.	–R		Chemical shift			
			Ipso	Ortho	Meta	Para
60	–NCO	Exp	5.1	−3.7	1.1	−2.8
		NN4	2.5	−2.1	1.2	0.8
		NN5	−2.1	−0.1	1.2	2.9
		NN6	−3.1	0.8	0.8	0.3
61	$-OCH_3$	Exp	31.4	−14.4	1.0	−7.7
		NN4	38.1	−16.2	1.2	−6.9
		NN5	38.1	−22.9	1.2	−12.2
		NN6	30.2	−20.1	1.2	−11.1
62	–OCN	Exp	25.0	−12.7	2.6	−1.0
		NN4	15.9	−9.3	1.2	−0.1
		NN5	21.7	−10.2	1.7	−1.1
		NN6	18.1	−8.4	1.2	−2.1
63	$-SCH_3$	Exp	10.0	−1.9	0.2	−3.6
		NN4	14.3	−6.2	0.8	−4.3
		NN5	17.2	−6.2	0.3	−3.7
		NN6	22.9	−9.3	1.2	−2.6
64	–Cl	Exp	6.3	0.4	1.4	−1.9
		NN4	10.8	−4.9	1.2	−0.1
		NN5	10.1	−6.2	1.2	−1.6
		NN6	−0.1	0.8	0.8	−2.1

factory interpretation of the meaning of the feedback connections and their potential importance for achieving the better fit of output activities with the required values.

REFERENCES

1 J.A. Anderson and Z. Rosenfeld (Eds.), Neurocomputing: Foundation of the Research, MIT Press, Cambridge, MA, 1989.
2 P.K. Simpson, Artificial Neural Systems, Pergamon, New York, 1990.
3 J. Zupan and J. Gasteiger, Anal. Chim. Acta, 248 (1991) 1.
4 D.E. Rumelhart and J.L. Cleland, Parallel Distributing Processes, Vols. 1 and 2, MIT Press, Cambridge, MA, 1986.
5 A. Lapedes and R. Farber, Physica (Amsterdam), D22 (1986) 247.
6 F.J. Pineda, Phys. Rev. Lett., 59 (1987) 2229.
7 M. Sato, Biol. Cybern., 62 (1990) 259.
8 V. Kvasnička, J. Math. Chem., 6 (1991) 63.
9 V. Kvasnička and J. Pospíchal, J. Mol. Struct. (Theochem), 235 (1991) 227.
10 F. Harary, Graph Theory, Addison-Wesley, Reading, MA, 1969.
11 E. Polak, Computational Methods in Optimization, Academic Press, New York, 1971.
12 H.-O. Kalinowski, S. Berger and S. Braun, ^{13}C NMR Spektroskopie, G. Thieme, Stuttgart, 1984.