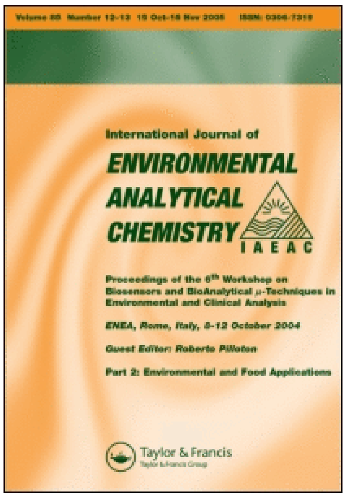

International Journal of Environmental Analytical Chemistry
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713640455

Artificial Neural Networks in water analysis: Theory and applications

Eleni G. Farmaki[a,b]; Nikolaos S. Thomaidis[a]; Constantinos E. Efstathiou[a]
[a] Laboratory of Analytical Chemistry, Department of Chemistry, University of Athens, Panepistimiopolis Zografou, 15771 Athens, Greece [b] Athens Water Supply and Sewerage Company (EYDAP SA), Water Quality Control and Protection Division, Polydendri Attikis, Greece

Online publication date: 27 January 2010

To cite this Article Farmaki, Eleni G. , Thomaidis, Nikolaos S. and Efstathiou, Constantinos E.(2010) 'Artificial Neural Networks in water analysis: Theory and applications', International Journal of Environmental Analytical Chemistry, 90: 2, 85 — 105
To link to this Article: DOI: 10.1080/03067310903094511
URL: http://dx.doi.org/10.1080/03067310903094511

Intern. J. Environ. Anal. Chem.
Vol. 90, No. 2, 15 February 2010, 85–105

Artificial Neural Networks in water analysis: Theory and applications[†]

Eleni G. Farmaki[ab], Nikolaos S. Thomaidis[a]* and Constantinos E. Efstathiou[a]

[a]*Laboratory of Analytical Chemistry, Department of Chemistry, University of Athens, Panepistimiopolis Zografou, 15771 Athens, Greece;* [b]*Athens Water Supply and Sewerage Company (EYDAP SA), Water Quality Control and Protection Division, Polydendri Attikis, Greece*

(*Received 2 December 2008; final version received 3 June 2009*)

Artificial Neural Networks (ANNs) have seen an explosion of interest over the last two decades and have been successfully applied in all fields of chemistry and particularly in analytical chemistry. Inspired from biological systems and originated from the *perceptron*, i.e. a program unit that learns concepts, ANNs are capable of gradual learning over time and modelling extremely complex functions. In addition to the traditional multivariate chemometric techniques, ANNs are often applied for prediction, clustering, classification, modelling of a property, process control, procedural optimisation and/or regression of the obtained data. This paper aims at presenting the most common network architectures such as Multi-layer Perceptrons (MLPs), Radial Basis Function (RBF) and Kohonen's self-organisations maps (SOM). Moreover, back-propagation (BP), the most widespread algorithm used today and its modifications, such as quick-propagation (QP) and Delta-bar-Delta, are also discussed. All architectures correlate input variables to output variables through non-linear, weighted, parameterised functions, called *neurons*. In addition, various training algorithms have been developed in order to minimise the prediction error made by the network. The applications of ANNs in water analysis and water quality assessment are also reviewed. Most of the ANNs works are focused on modelling and parameters prediction. In the case of water quality assessment, extended predictive models are constructed and optimised, while variables correlation and significance is usually estimated in the framework of the predictive or classifier models. On the contrary, ANNs models are not frequently used for clustering/classification purposes, although they seem to be an effective tool. ANNs proved to be a powerful, yet often complementary, tool for water quality assessment, prediction and classification.

Keywords: ANNs; back-propagation; Kohonen network; Radial Basis Function; water analysis; modelling; classification; chemometrics

1. Origin of ANNs

The development of ANNs was based on the Rosenblatt's [1] initial idea of a unit called perceptron, which produces an output of 0 or 1 (binary output) depending upon the weighted linear combinations of inputs. The structure of a single perceptron (single-layer neural network) is very simple: there are a number of inputs (x_n), weights (w_n), a bias (b)

*Corresponding author. Email: ntho@chem.uoa.gr
[†]Paper presented at the 6th Aegean Analytical Chemistry Days (AACD-2008), 10–13 October 2008, Karajayit, Turkey.

ISSN 0306–7319 print/ISSN 1029–0397 online
© 2010 Taylor & Francis
DOI: 10.1080/03067310903094511
http://www.informaworld.com

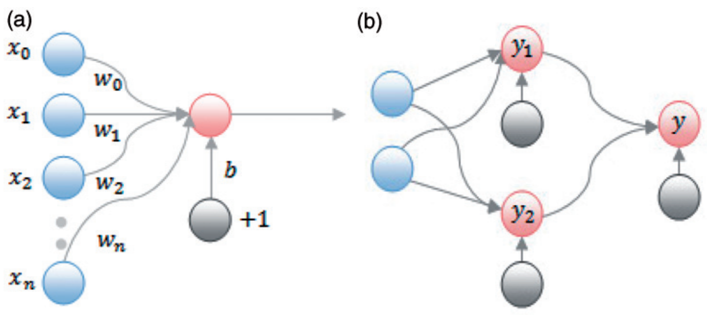

Figure 1. (a) The perceptron *vs.* (b) Multiperceptron Network [2].

Table 1. Training vectors for the three functions.

	Output		
Input (x_1, x_2)	OR function	AND function	XOR function
(0, 0)	0	0	0
(0, 1)	1	0	1
(1, 0)	1	0	1
(1, 1)	1	1	0

and an output (Figure 1a) [2]. For every input, a weight is applied (multiplied with the corresponding value), and the sum y is finally calculated by the formula:

$$y = \sum_{i=1}^{n} x_i w_i + b. \tag{1}$$

The presence of the weights in Equation (1) is critical, as they define the propagation of the signal along the network. Each signal that is propagated is also multiplied with the corresponding weight of the connection. The weights may be positive or negative. They reflect the degree of importance of every connection between inputs and output. They determine the correlation between the inputs data and the output information. Thus, it is considered that they conclude a whole knowledge of the network and reflect the necessary information to solve the problem. The initial selection of the weights is usually performed randomly. The weight adjustment is accomplished during the 'training' or 'learning' process (see Section 2.2), by the means of the learning rule. The training process of the network is the most important stage in ANNs development [3].

The bias is also considered as a weight (it is alternatively symbolised as w_o) of an input variable equal to the unit ($x_0 = 1$). With its application, another degree of freedom is added and the flexibility and versatility of the network is increased.

The perceptron could easily handle the classical OR and AND functions (Table 1). Linear separation of the data was easily achieved with the appropriate selection of the weights and bias. The learning rule of the perceptron is simple and could be described in four steps [4]:

(1) Start with random weights (w_i) and bias (b).
(2) Select an input x from the training set.

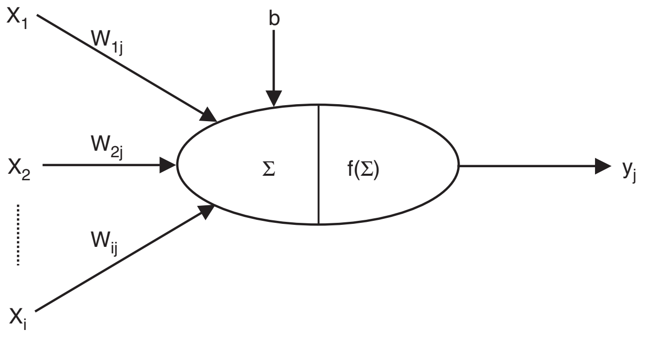

Figure 2. Scheme of an artificial neuron.

(3) If $y \neq d(x)$, where d(x) the theoretical response (output) for x, modify all the connections w_i and bias b, according to $\Delta w_i = d(x)x_i$ and $\Delta b = x_i$.
(4) Go back to step 2.

However, this type of network had limited abilities. The solution of exclusive OR (XOR) problem (Table 1) was not possible. The single layer model (input-output) could not separate the data using just a simple line. The solution was to add an intermediate (hidden) layer and the problem was now broken into three different linearly separable problems (Figure 1b). Multi-layer Perceptrons (MLPs) had been developed: adding hidden layers increased the class of problems that could be solved by the feed-forward networks.

2. Fundamentals

2.1 *General presentation of ANNs*

ANNs receive a number of inputs in the processing units which are capable of communicating by sending signals to each other over a large number of weighted connections. Some basic features of them should be initially emphasised [4]:

(1) A set of input signals $X_1, X_2, \ldots, X_i$.
(2) Connections for each unit. Each connection is defined by a weight w_{ij} between the unit i and j.
(3) An output y_j for each unit.
(4) An external input b (bias) for each unit.
(5) A propagation rule, which determines the effective input Σ from the external inputs x_i.
(6) An activation function f (usually sigmoid), which determines the correlation between the sum input $\Sigma = \sum_{i=1}^{n} x_i w_{ij} + b$ and the output y_j of this unit.
(7) A method (algorithm) for updating the information.

Figure 2 summarises the aforementioned features.

The main idea behind this is that the generated errors (differences between each output and the theoretical response) are propagated (distributed) along the units of the hidden layers. This is the back-propagation (BP) learning rule or algorithm. Back-propagation algorithm is considered as a generalisation of delta-rule (see Section 2.3) for non-linear activation functions and multi-layer networks.

2.2 *Training a network*

Working with a network means that the exact nature of the relationship is unknown. This relationship is established through the process of 'training' or 'learning'. There are two types of training used in ANNs: supervised and unsupervised [5]. In supervised learning, a set (pattern or case) of training data is used. This set contains input examples combined with outputs, in order for the network to 'construct' the relationship between them. The outputs are *a-priori* known and the network calculates and adjusts the weights (during some iterations or epochs) in such a way that the calculated and desired outputs are as close as possible [6]. This means that the prediction error in the training set is minimised through the appropriate adjustment of the weights. If the network is properly trained, it has then learned and the model can predict the outputs for unknown samples for given inputs through also an unknown function. A network with more weights models a complex function and is prone to 'over-fitting' or 'over-learning' [5]. This means that the network can memorise the training data and therefore be less able to generalise between other inputs and outputs. On the contrary, a network with fewer weights may not be powerful to model the data. The solution to the problem is the use of another sample set: the 'validation' or 'selection' set. This set checks the network performance from an independent view: as the training processes, the training and validation error are calculated. If the first is decreased and the second is increased, this indicates that the network starts to over-fit (Figure 3a). If, however, both training and validation errors drop, the network is trained properly (Figure 3b).

Unsupervised training means that the network is provided with a sample set and is left to settle down (or not) without a known desired output [6]. This network attempts to learn the structure of the data, recognising clusters of them (as with the Kohonen technique, see Section 3.5).

3. Algorithms

3.1 *Delta-rule*

A powerful concept in networks is the error surface. The N weights (and bias) are considered as dimensions in a space with $(N + 1)$ dimensions. The last dimension is the network error [5]. For any possible solution, the error must be calculated and the minimum in this surface must be found. In delta-rule and BP algorithms, the steepest descent minimisation method is used for this purpose. Thus, the gradient vector of the error surface is calculated. This vector points in the direction of the steepest descent, and as it moves along, the error is decreased. A sequence of such moves will end up to the minimum.

The delta-rule is a learning rule for single-layer neural networks. It is the base for the most important algorithm used in ANNs, the back-propagation (BP), which is described in the next Section (3.2).

As already mentioned, for a single-layer network with a linear activation function f, the calculated output y is given by the equation:

$$y = f(\Sigma) = f\left(\sum_{i=1}^{n} x_i w_i + b\right) \qquad (2)$$

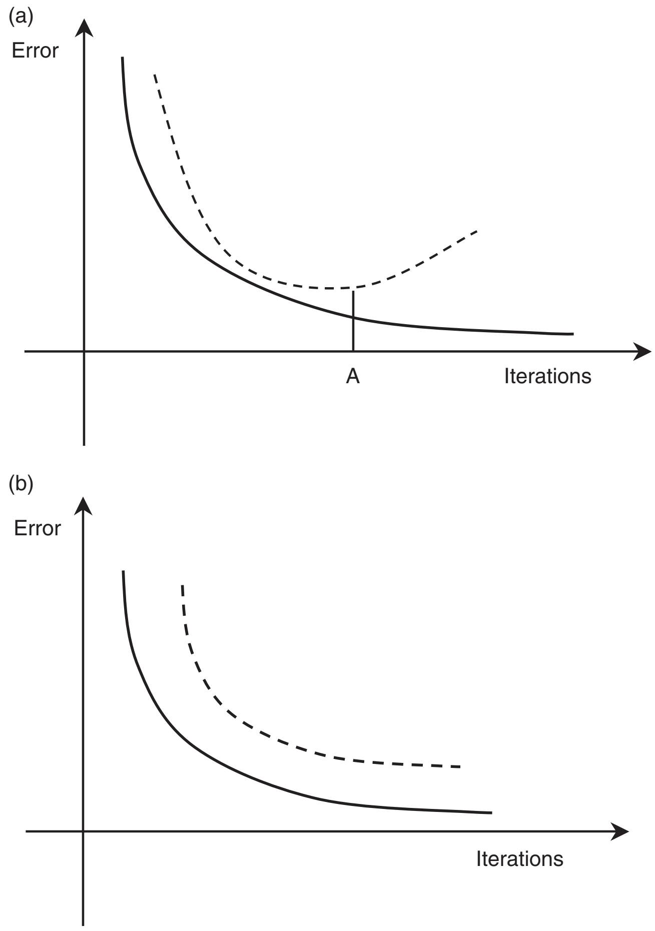

Figure 3. (a) Over-fitting of the network begins at point A. The solid line represents the training set, while the dashed line represents the validation set. (b) Ideal performance curve of the network [3].

Initially, the whole performance of the network must be measured and then this performance must be optimised [7]. The learning algorithm should change all the weights so that the output y^q, where q ranges over the whole set of input pattern, becomes more and more similar to the theoretical response d^q. The overall performance for the output nodes is measured by the formula [4]:

$$\mathbf{E} = \sum_{q=1}^{N} E^q, \quad \text{where } E^q = \frac{1}{2}(d^q - y^q)^2 \tag{3}$$

The idea is that function E must be minimised, with the appropriate adjustments to the weights w_i.

The gradient $\nabla \mathbf{E}$ of the function $\mathbf{E}$ at a point $\mathbf{w}$ with components w_i is the vector of partial derivatives $\frac{\partial E}{\partial w_i}$ [8]. Like the derivative of a function of one variable, the gradient always points to the uphill direction of the function $\mathbf{E}$. The downhill (steepest descent) direction of **E** is $-\nabla\mathbf{E}$. Thus, we move proportionally to the negative of $\nabla\mathbf{E}$, leading to the updating of each weight w_i as:

$$\left.\begin{aligned} w_i(t) &\rightarrow w_i(t-1) + \Delta^q w_i \\ \Delta^q w_i &= -n\frac{\partial E^q}{\partial w_i} \end{aligned}\right\} \tag{4}$$

where n is the learning rate. The terms t and $t-1$ refer to the present iteration and the previous iteration, respectively.

The derivative is [4]:

$$\frac{\partial E^q}{\partial w_i} = \frac{\partial E^q}{\partial y^q}\frac{\partial y^q}{\partial w_i} \tag{5}$$

and due to the linear function (Equation (2)):

$$\frac{\partial y^q}{\partial w_i} = x_i \tag{6}$$

while:

$$\frac{\partial E^q}{\partial y^q} = -(d^q - y^q). \tag{7}$$

From Equations (4) to (7) we have:

$$\Delta^q w_i = n\delta^q x_i \tag{8}$$

if we define $\delta^q = d^q - y^q$.

Thus, delta-rule adjusts the weights by checking the real and theoretical outputs. When the initial function **E** (Equation (3)) is minimised, the squared differences between the computed y^q and required values d^q are zero. Consequently, the delta-rule leads to the search of the weight vector **w**, where the error function **E** is minimised through its zero derivatives. Delta-rule can be applied in continuous and binary inputs and outputs.

3.2 *Back-propagation*

From the XOR function example, it became obvious that, for most of the problems, multi-layer networks are needed, for approximating general relationships. Algorithms for these more complicated neural networks are also necessary [8]. The major problem was that, while it was easy to define the theoretical response for the output neuron, it was difficult to define the desired output for the hidden layer [3]. The following popular learning algorithm, referred to as the back-propagation algorithm (BP), was first published by Rumelhart *et al.* [9]. It was used to train these Multi-layer Perceptrons (MLPs) and is considered as a generalisation of the delta-rule discussed above. When a sample pattern is introduced in the network, the activated values (through weights, bias and an activated function) are propagated to the output units. The squared differences between the computed and required (theoretical) values are calculated first. In order for these values to be set to zero, the simplest method is merciless [4]: all the weights must be adjusted.

The error term δ_k^q for the observation q (one of the whole pattern) in the output neuron k is given by:

$$\delta_k^q = (d_k^q - y_k^q)\, y_k^q (1 - y_k^q) \tag{9}$$

where the term $y_k^q(1 - y_k^q)$ is the derivative of the sigmoid function [10].

The error term at the node j of the hidden layer that uses a sigmoid function is:

$$\delta_j^q = y_j^q (1 - y_j^q) \sum_{k=1}^{K} \delta_k^q w_{jk}. \tag{10}$$

This is the first step. With this way, however, not all the weights are changed. So the next step that gives the solution to this problem is the distribution of the error from the hidden layer to the input one. This approach justifies the name of the algorithm: the output error is propagated backwards and distributed from the output layer through the hidden layers to the input layer. The error terms from the output are back-propagated through the network by making adjustments to the weights till the initial inputs. These adjustments are calculated according to the formula:

$$\Delta w_{ij}(t) = n \delta_j^q y_j^q + a \Delta w_{ij}(t-1) \tag{11}$$

where Δw_{ij} is the update in weight between the node j in the hidden layer and the node i in the input layer. In Equation (11), n is the learning rate ($n > 0$) and a is the momentum.

The propagating behaviour can be clarified through the Equations (9) to (11): initialising from the calculated error $d_k^q - y_k^q$ in the output layer, the correction (adjustment) passes to the weight w_{jk} of the hidden layer and ends to the input through its weight w_{ij}. We can imagine a domino game that moves forward to the outputs, but then unexpectedly, returns back and transmits the movement back to the inputs.

The terms t and $t - 1$ refer to the present and previous iteration, respectively, as in the delta rule. The momentum a is a coefficient used to make the new weight dependent to the previous one [4]. Thus, when learning rate n is large, the oscillations can be avoided and the minimum is reached (Figure 4). The use of momentum may also accelerate the final result (see below).

Back-propagation learning way may proceed in two ways: pattern (or case-by-case) mode and batch mode. In the first mode, the calculations are performed after each pattern (case), while in the second mode, the calculations and subsequently the weights updating are performed after the whole training pattern is presented [6].

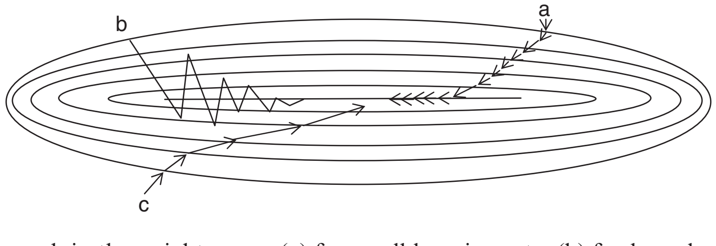

Figure 4. The search in the weight space: (a) for small learning rate, (b) for large learning rate with many oscillations, (c) with large learning rate and momentum [4].

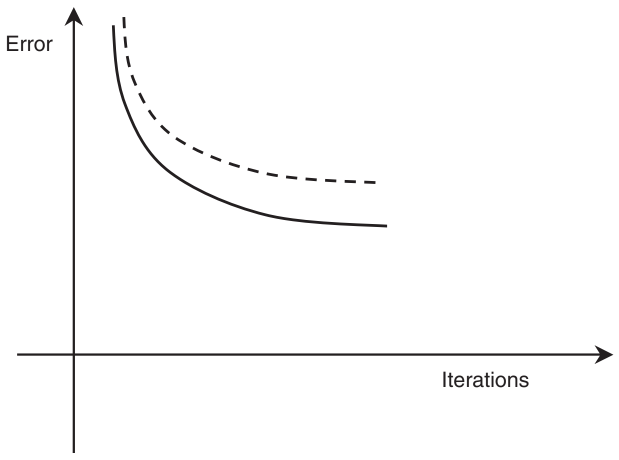

Figure 5. 'Paralysed' network. The error for the training (solid line) and validation set (dashed line) is too high to be accepted [3].

Back-propagation has lower memory requirements than other algorithms [5]. It is a good choice if the data set is very large and contains a great deal of redundant data: doubling the cases data, the convergence time will be longer, but the results will be the same. However, it can also be equally good if the data set is small: a more sophisticated algorithm would be stern about the size of training and validation sample set. Moreover, the algorithm may be modified by the use of the momentum term. This makes the algorithm to speed if it moves for several steps in the same direction. The result is quicker convergence to the error function minimum and local minima skip.

Despite the advantages described above and the success of back-propagation (apparent from the numerous applications), some disadvantages have been detected and finally led to optimised versions of it by a number of researchers (see Section 3.3) [4]:

- The network is paralysed (Figure 5). During network training the weights may be adjusted to very large values and because the usual activation function f is the sigmoid one, the result will be close to one or zero. Subsequently, noting that:

$$\text{if } y = f(S) = \frac{1}{1+e^{-S}} \rightarrow f'(S) = y(1-y) \tag{12}$$

  the training process can come to a standstill.
- The algorithm may lead to a local minimum, when there is a deeper region nearby.
- Some problems are slowly converging [6] and the time for training the algorithms is really high.

3.3 *Back-propagation modifications*

Two basic modifications of BP algorithm are the quick-propagation (QP) and the Delta-bar-Delta.

Quick-propagation is a batch mode algorithm: the weights are updated after all cases are performed. This signals the end of the iteration. Quick-propagation (QP) starts with the same rule as back-propagation (BP), but continues based on the assumption that the error-surface is locally quadratic, so if this is the case, the algorithm can converge on the minimum very rapidly [5]. Problems arise when the error surface is not concave and the algorithm could go the wrong way. Generally, QP seems more inclined to local minima than BP and it tends to be unstable.

Delta-bar-Delta algorithm can also be inclined to local minima, but it is quite stable compared to QP algorithm. It is also a batch mode algorithm and it is inspired from the observation that the error surface may have different gradient along each weight direction; so each weight should have its own learning rate. Consequently the individual learning rates are updated for each weight on each epoch, keeping in mind, that two basic demands should be fulfilled:

(1) If the derivative has the same sign for several iterations, the learning rate is increased (the direction seems to be right and it is wise to keep sloping the same way);
(2) If the sign derivative alternates for several iterations, the learning rate is decreased.

In order to satisfy the above requirements, Delta-bar-Delta algorithm has an initial learning rate, used for all weights on the first epoch, but then an increment factor is applied to them, when the derivative sign does not change, or a decay rate is multiplied when the derivative sign changes. The risk here is that the algorithm could be prone to noisy error-surfaces.

3.4 *Other algorithms derived from back-propagation*

Conjugate Gradient Descent is an advanced algorithm that can be used wherever back-propagation is applied. It is a batch mode algorithm and it is recommended for any network with large number of weights and/or multiple output units. Conjugate Gradient Descent starts with the direction of the steepest descent as BP does, but continues by creating a series of lines that search across the error surface. The algorithm projects a straight line in the aforementioned direction and then locates a minimum along that line, a process that is quicker, as it involves searching in one dimension [5]. The directions of the searching lines are chosen to ensure that the already minimised ones stay minimised. If the algorithm discovers that the current line search is not actually downhill, it calculates the line of the steepest descent and restarts the search in that direction. Thus, each epoch of the Conjugate Gradient Descent algorithm may involve one calculation and many error evaluations that consequently increase the real time needed. So, an epoch may be 3–10 times longer than a BP epoch. However, Conjugate Gradient Descent has low memory requirements, proportional to the number of weights.

Quasi-Newton is also an advanced algorithm that can be performed wherever BP is applied and generally with better results. It is a batch mode algorithm and is recommended for networks with small number of weights [5]. Quasi-Newton starts with the direction of the steepest descent as BP does, but continues by building up an approximation to the inverse Hessian (matrix of second partial derivatives). For small problems, where the second derivatives are easily calculated, the method is efficient, but it is problematic in larger problems [6]. Quasi-Newton has high memory requirements proportional to the square of the number of weights; it may be inclined to local minima and is less stable than

Conjugate Gradient Descent. However, Quasi-Newton is considered fast convergent compared to the Conjugate Gradient Descent algorithm.

3.5 *Radial Basis Function networks*

Like the MLPs network, the Radial Basis Function (RBF) network consists of three layers: input, hidden and output. Each input neuron is connected to all the hidden ones, while hidden and outputs are interconnected to each other by a set of weights [11]. The hidden layers transform the input data using a function that is usually a Gaussian. This symmetric function has a centre μ and a spread σ. The spread which is the radial distance from the centre is different from zero. For each input that is distant from the centre (due to the Euclidean distance [12]), the Gaussian function decays the result to zero. Thus, the function responses only to a small input space around its predefined centre.

The key for a successful RBF network is the appropriate choice of the centre and spread [3,13]. Moreover, by means of these, RBF networks divide the input space in hyperspheres, whereas MLPs use hyperplanes [3].

Finally, each unit in the output layer makes a linear transformation to the data of the hidden layer. The number of units in the hidden layer is established during the training process: in the beginning this number is zero, while hidden neurons are added until the stopping rule is fulfilled.

The RBF networks have some advantages compared to MLPs, so that they sometimes seem to be more attractive than the latter:

(1) They are generally more robust and reliable in ‘noisy’ data [12].
(2) The adjustment of the weights of the output layer is considered relatively simple due to the liner correlation and the training process guarantees a certain convergence [5,13].
(3) The network convergence and training is achieved faster, as there are no local minima at least in the output layer [5,14].
(4) They provide good ‘generalisation’ with the minimum number of neurons [14].
(5) They require lower *samples to variables ratio* than the MLPs networks [15].

However, the selection of the appropriate centres is pivotal in the RBF network design and comprises an obstacle to its preference [5,14]. However, this can be solved by combining RBF networks with self-organising maps (SOMs, see next section) [14]. Additionally, the RBF networks require more hidden units to approximate sufficiently a function [5]. Consequently, an RBF solution will tend to be slower to execute and more space-consuming than the corresponding MLP, even if it is much faster to train. Extrapolation far from training data is forbidden: the response for such inputs is zero [5].

3.6 *Kohonen neural networks*

Kohonen neural networks (also referred as self-organising maps, SOMs) are considered as an unsupervised technique that attempts to recognise clusters within the training cases. It is a learning procedure that identifies some pivotal points in the space (characterised by weights) and maps the samples onto a two-dimensional layer according to the proximity to these points. Each point is characterised by a weight vector which is associated (calculating the between distance) with the respective vector of the patterns (cases). The point that is closer to the pattern is the winner: its weights are updated and the pattern is considered to 'belong' in its cluster. The new weights are calculated from the old ones through a learning rate, initially defined and updated (usually geometrically), after each iteration. Moreover, having initially defined the so called 'neighbours' of the winning cluster through a radius R, more weights points can be updated after the association of each pattern. The larger is the R, the more neighbor clusters update their weights. The value of R is reduced after learning rate update. The algorithm could be summarised in seven steps [16]:

(1) Initialise weighs w_{ij} for each cluster unit j, set radius R and learning rate a.
(2) For each pattern vector $\mathbf{x}$ (x_i) do the steps 3 to 5.
(3) For each unit j calculate the distance $D(j) = \sum_i (w_{ij} - x_i)^2$.
(4) Find the unit j that has the minimum distance (the winner).
(5) For the winner j and the neighbours (defined by the R), update the weights by the formula: $w_{ij}\,(\text{new}) = w_{ij}\,(\text{old}) + a\,[x_i - w_{ij}\,(\text{old})]$.
(6) After all patterns tested, update learning rate and radius.
(7) Test stopping condition (this involves the number of iterations) [15].

Training is usually performed in two phases: initially, a first rough one with a large neighborhood radius and finally, a second one, more detailed (fine tuning), with a small radius. This process results in classifying the data in groups with which they have the most similar characteristic vectors (variables for the inputs – weights for the points that characterise these groups) [17]. Kohonen [18] discriminates the two aforementioned phases in this way:

(1) the initial formation of the right order and
(2) the final convergence.

The second phase lasts much longer and requires a small learning rate.

There are many alternative architectures for Kohonen networks: maps with no structures, with linear structure, or two-dimensional topologies for better visualisation.

One of the main drawbacks of Kohonen maps is that the performed mapping is discrete and the number of available mapping positions is limited by the dimension of the two-dimensional Kohonen layer [15].

4. Applications of ANNs in water analysis and classification

For the preceding overview of the theory of ANNs, it is apparent that ANNs are a versatile technique offering different architectures and training algorithms that could be applied to a wide range of problems. Concerning the application of ANNs in chemical analysis, few review articles have been published during the last ten years; however, none has presented an overview of the ANNs applications in water quality assessment. Initially, in 1997, Svozil *et al.* [6] published the first tutorial article for MLPs, including application examples in spectroscopy, process control, protein folding, QSPR (Quantitative Structure Property Relationship) and general analytical chemistry. In the same year, Zupan *et al.* [19] in their tutorial and pioneering work for Kohonen and counter propagation (CP) networks, reviewed some applications of ANNs in analytical chemistry, such as classification, modelling and prediction using spectroscopic data. Ten years later, four new review articles highlighted the explosion in application of ANNs in environmental, food and analytical chemistry. In 2008, Kalteh *et al.* [14] reviewed the applications of

Kohonen networks in various hydrological processes such as river flow, rainfall run-off, precipitation and surface water quality issues. A year earlier, Berrueta *et al.* [20] presented the mostly used pattern recognition techniques used in food analysis, for exploratory analysis, variables' selection and reduction and supervised pattern recognition, including PCA, CA, LDA, PLS, CART and, eventually, ANNs. In 2008, Marini *et al.* [7] presented different artificial neural networks (MLPs, Kohonen and CP networks) and their application in food analysis and for different kinds of chemometric problems (exploratory analysis, unsupervised clustering, classification and regression analysis). Most recently, in the beginning of 2009, Marini *et al.* [15], in a comprehensive review, presented several examples of ANNs applications in food analysis, including RBF networks applications.

In this section, recent applications of ANNs in water analysis and water quality assessment will be described. ANNs (mainly MLPs, RBF and SOM optimised models) are often used as a tool for *modelling and prediction* [10–12,21–37], applied to *water quality assessment* [37–39,46], sample *classification* [38–45], or for exploring variables' *correlation and significance* [30,31,35,37–39,42,44], frequently compared to more convectional statistical techniques [22,26,27,36–39,44]. Moreover, traditional PCA (Principal Components Analysis) scores are often used as the inputs, usually when the initial variables are numerous [10,21,36]. Overall, ANNs seem to produce better results than the traditional chemometric techniques and fulfil the authors' expectations. Additionally, the theoretical and experimental background included in these studies comprises a valuable basis for an overall evaluation of ANNs architectures through a critical comparison of advantages and disadvantages that are finally discussed below.

4.1 *Regression and prediction*

In this section the works that used ANNs for model construction and prediction are reviewed. Supervised ANNs (mostly MLPs-BP algorithms) are particularly useful when the relation between variables and predictors is not linear and noisy data have to be treated.

Kompany-Zareh *et al.* [10] studied the simultaneous determination of Fe and Ni in aquatic mixtures. The PCA scores were calculated from the absorbance values of the complexes of Fe(II) and Ni(II) with xylenol orange, and they were used as the external inputs to the network. Thus, with the help of PCA, the data were reduced without any loss of information. Back-propagation was used as the learning algorithm in the MLPs network and a sigmoid function was applied. The network was trained with a series of calibration standards and the best parameters (learning rate, momentum, number of nodes in hidden and input layer) were chosen based on the minimum prediction error. The constructed model was used for the prediction of Fe(II) and Ni(II) in real samples (wastewater from industries). The results for the two metals (real values measured with FAAS and the predictive ones) showed good agreement [10].

Different ANNs architectures were explored for the prediction of NO_3-N concentration in drainage water in another study, published in 2003 [11]. This information could be used for proper fertiliser management. Thus, different ANNs models were trained, tested, optimised and compared. Eight inputs, such as field treatment (conventional tillage or no tillage), total nitrogen applied, and rainfall per day, were fed to the ANNs (MLPs-BP network and RBF). The available data were divided into training and test samples and the models were evaluated through parameters such as the correlation coefficient between predicted and observed values. The MLPs network was optimised by varying two parameters: learning rate and the number of hidden nodes, while in RBF, the tolerance (which determines the number of the hidden layers) and the spread were optimised. Overall, RBF was proved better than MLPs-BP in predicting NO_3-N concentration on both daily and monthly basis, while MLPs could be used only on monthly basis.

Comparative studies between ANNs architectures were carried out, targeting in the prediction of pesticide occurrence in rural domestic wells from limited information [12]. Pesticide prediction in the wells was accomplished using six costless input parameters such as the well depth and the time of sample collection. Multi-layer Perceptrons with BP algorithm (topologies with one and two hidden layers) and RBF were applied (among others) in the data and comparative studies were carried out. The input variables were categorised due to their degree of pesticide occurrence. Thus, swallow wells (more vulnerable to contamination) were represented by a higher value in the ANN model. Similarly, output variables (pesticides sum concentration) were categorised as high, moderate, low and none. For the models evaluation, four parameters were set: a correlation coefficient between predicted and observed values, the corresponding root mean square error, the model performance due to its predictive ability and the matching of the final classes. Training and test was performed by a total of 69 samples. The poor prediction ability of the RBF model (with one hidden layer) ascribed to its three-layer structure and the large number of neurons in the hidden layer. Thus, the generalisation ability was lost. Similarly, the MLPs-BP model with one hidden layer could not generalise and the addition of hidden neurons could improve the training performance, but decline the prediction ability for new samples. On the contrary, a BP model with two hidden layers (6-6-4-1) seemed to be the most accurate. The numbers represent the nodes for every layer: 6 inputs, 6 nodes in the first hidden layer, 4 in the second and 1 output. The model achieved an 89% prediction accuracy of pesticide occurrence. Moreover, the predicted values were less scattered around the observed (theoretical) values. Sensitivity analysis was also carried out and the necessity of using all the input variables was confirmed [12].

Safavi *et al.* [21] exploited the different kinetic behaviours of Co(II) and V(IV) during the reaction with Fe(III) for the production of Fe(II), which forms a coloured complex with the chromogenic reagent 1,10-phenanthroline, aiming at the simultaneous determination of Co(II) and V(IV). To accomplish that, a series of synthetic mixtures was used for training and testing (constructing and optimising) the ANNs. Thus, the learning rate, the number of principal components (PCs) used (absorbance data) and the number of hidden nodes was successfully evaluated based on the minimum error of prediction. The quality of the network was further tested by an external validation sample set. Finally, real water (mineral and tap) samples were used and the results (good agreement between actual and found concentration values) confirmed the powerful applicability of ANNs and the proposed kinetic analytical method. Emphasis was given in the fact that, due to synergistic effects, non-linearity was a one-way solution and that was efficiently provided through ANNs. On the contrary, linear chemometric methods were unsuitable for solving this problem [21].

The comparison between Principal Component Regression (PCR), Partial Least Squares (PLS) and ANNs for the analysis of mixtures of polycyclic aromatic hydrocarbons (PAHs) was the aim of the next study [22]. Water samples that have been spiked with ten PAHs were used for the models evaluation. Pollutants determination was performed by synchronous fluorescence. Back-propagation was used as the learning algorithm in ANNs and the network was trained efficiently by calibration standards with a wide concentration range. Parameters such as the number of the nodes in the hidden layer, the learning rate, the type of the function used, were efficiently optimised. Overall, the PLS method seemed to be the best for the prediction of PAHs concentration. However, in some cases the differences with the other techniques were not high. Results obtained by ANNs or PCR were roughly equivalent, although the prediction ability of each system depended on the individual component determined. Time consumed was much shorter in the case of ANNs [22].

A MLPs-BP model was used in order to establish a model for evaluating the change in electrical conductivity (EC) and dissolved oxygen (DO) values in recharge and discharge areas of the dam water under pollution risks [23]. Specifically, physical and chemical parameters have been measured in selected sampling sites that represented recharge and discharge areas in the lake Mamasin, in central Turkey. Six neurons were included in the input layer, stood for the sampling periods, the sites (recharge or discharge), suspended solids, total N, temperature and rainfall data. Two hidden layers were used in the optimised BP model and two output neurons (values of EC and DO) were derived. The comparison with the real measured values showed good agreement [23].

Two different modelling techniques with different tasks were used in the next study [24]. The authors performed regression analysis to correlate TOC with BOD and TOC with COD in water samples, collected from the effluent treatment plant of a refinery in India. A few other parameters (TSS, TDS, phenol, NH_4-N and Kjeldahl's N) were also measured in the same samples and used as inputs in ANNs-BP models. Finally, a total of 12 models were developed to predict BOD, COD or both. Parameters such as the number of input parameters, the hidden neurons and the learning rate were optimised and the best model (with the lowest error) was chosen for each case. The results (agreement between measured and predicted values) in all the final models were very good.

Aiming at studying the uranium distribution in surface water samples [25], Kohonen algorithm was used to visualise origin (streams, dams and wetlands) and trends. According to the authors [25], Kohonen SOMs seemed to have potential not only for predicting and characterising distribution trends, but also for risk assessment studies, combined with geological speciation modelling.

Targeting in estimating the extent of As contamination in the groundwater of Malda district, India, Purkait *et al.* [26] used MLPs-BP in a recent study published in 2008. Parameters such as pH, conductivity, total dissolved solids, salinity, dissolved oxygen (DO), were used as inputs, while As concentration in groundwater was the evident output. Comparison of the final four-layer feed-forward back-propagation ANN model with conventional methods (multiple regression analysis and active set support vector regression) showed that ANNs were the best method for As concentration prediction.

Multiple regression (MR) and ANNs models were developed and compared to support the water treatment plant operations in Geum river of Korea through providing forecasted NH_3-N concentrations [27]. It was shown that the NH_3-N concentration for the next time step was dependent on dam outflow, alkalinity, temperature, and NH_3-N concentration of the previous time step. Finally, while in the learning phase, the ANN models compared to MR models showed a better agreement between predicted and observed data, during the second phase of verification, the performance of ANN models was decreased and provided similar results with MR models. It was concluded from the model comparisons that both ANN and MR models should be used to avoid overestimation of the NH_3-N concentrations up to 3-lag months during low flow season [27].

Other similar examples also concerned the prediction of nitrogen compounds (usually nitrate) in rivers and groundwater [28–30], BOD in surface water samples [31], or dissolved oxygen in drain canals and rivers [32].

Moreover, the salinity variations of Apalachicola river in USA has been successfully assessed from tides, winds and river flow as inputs, by means of a ANN-BP model, instead of hydrodynamic and transport models [33], polycyclic aromatic hydrocarbons have been quantified in water samples in Spain [34] and pesticides' concentrations have been predicted in groundwater [35] or in different water samples (tap, lake and pond water) [36]. Particularly, in the last work, ANN-RBF model leads to better performance compared to traditional techniques such as principal component regression (PCR) and partial least squares (PLS). The authors [36] had to handle serious overlapping problems of voltammetric peaks of four carbamate pesticides and needed flexible, different calibration models to facilitate their resolution and simultaneous prediction. The final RBF model was the most effective on the basis of the predetermined criteria. The authors emphasised the significance of the variables reduction during the pre-treatment with PCA, as background scatter and noise were not modelled any more.

4.2 *Water quality assessment through ANNs modelling and prediction*

The concentration of several pollutants (from metals to PAHs) in wastewater samples was predicted by neural network data processing of absorption and fluorescence measurements [40]. Back-propagation was used as the learning algorithm in a linear network and synthetic mixtures were generated for calibrating and test purposes. Moreover, optical fibre instrumentation for absorption spectroscopy and a flow cell for fluorescence measurements were proposed. The results (quantitative predictions of component concentrations of mixtures) were actually accurate.

Wastewater reclamation was evaluated [41], by applying a neural network model as a forecasting tool to facilitate the decision-making process in effluent reuse applications. A number of input variables of the sample influent such as pH, DO, BOD, ORP (oxidation/reduction potential) were applied to the BP neural network and a sigmoid function was used as the transfer function. The three outputs (predicted values) concerned the NH_4-N, NO_3-N, and total nitrogen of the sample effluent. The model was trained by a pattern of 67 cases and the input and output variables were normalised to a range between 0 and 1. Network parameters such as the number of hidden layers and the number of nodes in the hidden layer were optimised based on the criterion of the MSE (mean square error) minimisation. A sensitivity analysis was carried out in terms of predicting NH_4-N and total N by exploring the effect of including or not the input variable of ORP in the model. The accuracy of the model was the criterion of this analysis and ORP proved to be a critical parameter, as accuracy decreased dramatically, when it was excluded of the model. Finally, a rainfall index was provided by the authors to 'correct the results' and adjust the criteria for the effluents released due to the differentiation of the dry and wet season. According to authors, the neural network model selected could be used as an on-line numerical tool for predicting effluent N, or as a useful tool providing information of the potential impacts of a reuse action on the aquatic environment [41].

Kohonen network was used to analyse the non-linear relationships among surface water quality parameters [42]. The purpose of that study was to evaluate the complex non-linear relationships between water quality parameters, using the SOM architecture.

Sixteen water quality parameters were measured in surface water in Turkey and the results were used to create a component map in order to understand the interdependencies between the initial variables. The number of neurons was determined experimentally to give the best compromise between the visual clarity and computing time. As dependencies between variables were revealed, SOM networks proved to be an efficient way to develop management tools, in order to solve agricultural, industrial and municipal water pollution problems.

Aiming at assessing the seawater quality in a Singapore coast, Palani *et al.* [43] used optimised MLPs-BP models to predict salinity, water temperature, dissolved oxygen, and chlorophyll-α concentrations. The ANNs models showed very good correlations between observed and predicted values and the authors despite the limited available sample size, considered them as tremendous potential forecasting tools.

Five parameters such as COD, Pb, suspended solids, total Kjeldahl nitrogen and total phosphorus were used iteratively as inputs in a MLPs-BP model optimised in the next study [44]. The purpose of this work was the assessment of storm water quality by prediction of the aforementioned variables one by one. However, regression models seemed to be more accurate and less time consuming than the ANNs models, when unknown samples were tested. Thus, the authors concluded that ANNs models were less applicable than regression models at predicting urban storm water quality. Moreover, sensitivity analysis was conducted to determine the relative significance of variables used in the ANN models and identify differences between ANN and regression models. The relative coefficients for the two methods were generally comparable, but regression models were more explicable, while ANN sensitivity analysis was considered limited.

Another example of ANNs application include the prediction of water quality in the water reservoir of Te-Chi in Taiwan [45]. Four models have been constructed concerning the prediction of eutrophication indicators such as TP (total phosphorus), DO, SD (secchi disk depth) and Chl-α (chlorophyll-α). The potential inputs variables for each of these models were pre-screened from previous studies. Trial and error processes and models training were also used to evaluate the results. Thus, the TP model was built by the input of PO_4^{3-} and SS. Inputs of the DO were month, pH, Chl-α, NH_4^+, NO_3^-, and water temperature, while the inputs of SD model included month and SS. The Chl-α model was constructed by 8 inputs: month, water temperature pH, DO, SD, SS, NO_3^-, and PO_4^{3-}. In all four models, the correlation coefficients between the observed and predicted values were above 0.7. Thus, according to the authors, the complex mechanism of eutrophication in Te-Chi reservoir could be quantified and expressed successfully by these four ANNs models and effective water management could be achieved.

4.3 *Classification*

ANNs have not been widely used for water sample classification and only recently, mostly Kohonen networks have been applied to such problems.

Principal components analysis, Kohonen neural network and counter propagation artificial neural network (CP-ANN) were applied for clustering and classification of a series of river water samples according to their pollution condition [37]. Moreover, the work aimed at finding correlations between biological classes and variables obtained by chemical measurements. Variables such as the river flow, water temperature, DO, COD, BOD5, pH, NO_2^-, NO_3^-, absorbable organic halogens (AOX), were weakly correlated and so PCA was not discriminant enough. Thus, less than 50% of the variance was explained with the first two components. On the contrary, ANNs models were more suitable for clustering all the 207 water samples as well as for the prediction of biological classes (from moderately polluted to heavily polluted). Sensitivity analysis accented AOX as the parameter with the greatest discriminant power [37].

The aim of an important recent study was to compare objects or variable classification by different techniques for large, non-simulated environmental data set. Three multivariate techniques, traditional (such as cluster analysis (CA) and PCA) and modern (such as SOM networks), were compared [38]. The data concerned 14 chemical parameters and 23 sampling sites in a river in Bulgaria, while two approaches were tested: monthly and annual average concentration values. Due to different sampling periods 1104 cases were finally indentified. Finally, PCA, CA and SOM produced approximately the same results for classification of the variables. On the contrary, due to the large number of objects, the PCA score plot and CA dendrogram were not appropriate to classify and detect groups into the sampling sites. However, SOM offered a more reliable classification and visualisation option. According to the authors, SOM networks proved to offer a special resolving power compared to the traditional multivariate techniques [38].

Kohonen networks were also used in the next study of the same research group dealt with the interpretation of a large monitoring data set for surface water quality [39]. The authors aimed at demonstrating how the more advanced multivariate techniques could contribute to better understanding of the data, by revealing similarities between sites or variables and by documenting seasonal variations. Thus, N-way PCA was performed for detection of spatial and temporal patterns. Tucker 3 modelling was proved to be more informative than the classical PCA. Moreover, as it was already discussed in the previous work, SOM network proved to be very efficient in revealing correlations between variables by the visualisation of the distance between them. The clustering of the sampling sites was achieved in a 9×12 Kohonen map, reflecting intrinsic similarities. According to the authors, the combined application of the two techniques was proved to be an effective and promising tool for handling a large environmental data base [39].

Counter-propagation artificial neural networks (CP-ANNs) were used by Grošelj *et al.* [46] for bottled mineral water classification, due to their geological origin. Major ions and pH were used as inputs and the sample origin was well predicted.

5. ANNs: An evaluation

It is evident by the aforementioned review that most of the ANNs works are focused on modelling and parameters prediction. In order to assess the water quality of water bodies, reservoirs and processes, extended predictive models are constructed and optimised, while variables correlation and significance is usually estimated in the framework of the predictive or classifier models. On the contrary, ANNs models are not frequently used for clustering/classification purposes, although they seem to be an effective tool. Thus, new research should be encouraged towards using ANNs models in classification problems.

Artificial Neural Networks can overall be assessed through the numerous works that concern the parallel and comparative use of these models and the more traditional chemometric techniques. Conclusions can be derived from the ANNs practical implementation, but from the theoretical presentations, as well.

Initially, the majority of researchers conclude that ANNs models prove to be effective tools fitting their purpose. Minor exceptions found ANNs techniques less accurate or difficult to implement [27,44], compared to traditional ones.

A number of advantages can also be derived from the theoretical sections:

(1) Through the process of training or learning, ANNs, are capable of 'capturing' any information hidden in the data and use it (by carefully adjusted weights) in the unknown samples without any *a-priori* knowledge for the correlations between inputs and outputs [6,12,19–21,30,34].
(2) The whole process for finding the best model assimilates the fewer variables with the most information [12,30].
(3) ANNs encourage non-linear models for the input data so that the most efficient approximation to the real function is achieved [5,6,12,15,20,21,45,47].
(4) They seem to be robust in 'noise', providing accurate predictions, regardless of uncertain data or measurements errors [5,6,10,20,48].
(5) Simultaneous calculating steps imply fast data processing and higher tolerance in problematic hardware support [20].
(6) Training, the possibility of feedback coupling of outputs with inputs, easy use and networks adaptation allow models' update by changing their internal structure according to the 'environment' changes [5,12,15,19,20,48].
(7) The generalisation ability allows the application of unknown data in the already trained model [5,6,12,20,48].
(8) Theoretically, they can approximate any function, mainly the MLPs [12,15,43].
(9) Their operation is based on less assumptions and restrictions for the data distribution, compared to the traditional techniques [15,19].
(10) The great variety of models (architectures and algorithms) offers many different alternatives and possibilities for problems' solutions.
(11) A number of ANNs techniques may be performed independently, exploiting and selecting variables, constructing and validating the model, without the assistance of the other multivariate techniques [6,12,21].

Specifically, Kohonen networks may project variables and objects in a two-dimensional space preserving the topological structure, while in the case of linear PCA, this does not always prove to be successful [15,38]. Since Kohonen networks are a non-linear mapping technique, there can be cases where they provide a clearer separation among the samples than PCA. Moreover, Kohonen networks can perform modelling, providing that a capable number of training samples is available. Compared with CA, Kohonen networks can be applied as a supervised technique and classify new samples. Thus, even if the two methods may end to the same clusters, SOM models differentiate by providing prediction ability for new inputs.

The drawbacks of ANNs are:

(1) The final solution depends on the initial status (e.g. the initial weights) of the network. If the initial weights are not appropriate, the network may never succeed

the desired goal, regardless the number of the epochs. So, training and validation processes seem to be entirely depended on the initial weights [12].

(2) The optimisation procedure of the ANNs models is frequently time-consuming and a lot of parameters must be evaluated in order to find the best model.
(3) It is important that the relative correlation of the inputs is known. Many combinations of them should be tested, as variable abundance may induct to network over-fitting [12,49].
(4) The final result is more or less a 'black-box'. The user must really guess what is 'hidden' inside a successful model [47], without any theoretical interpretation [6]. The user does not really know if the training and validation sets were representative [3].
(5) There are usually many local minimum in the error surface. It is impossible for someone to guarantee that the whole minimum was really approached [3].
(6) Extrapolation is usually dangerous and unjustified.
(7) The networks are not sometimes adaptive in new data and so retraining is demanded.
(8) A large number of samples is needed for network training. The validation of the results must also be evaluated carefully [3].

Concerning the last topic, it seems that the most important aspect for successful ANNs models is the *variables to samples* ratio. Moreover, the variables reduction is critical for the development of the ANNs models. It is usually achieved by data pre-treatment by means of traditional techniques such as PCA, DA, or even Pearson correlation. The presentation and the use of all the available variables in the networks construction may cause problems. For over-fitting avoidance, the '*principle of parsimony*' must be kept. Complex models do not generalise, but tend to focus on the training sample set, modelling the system noise, as well. Thus, among equivalent models, the one with the fewer variables or the simplest architecture must be chosen [20,34]. Some disadvantages of the models with two many variables are:

(1) large computational complexity and memory requirements [14];
(2) increased requirements for samples number [5];
(3) difficulties in training [14];
(4) misconvergence and poor models performance [14,50];
(5) increased complexity of the models, over-fitting phenomena, difficulties in understanding;
(6) increased 'noise' due to the inclusion of non-significant variables [14].

Thus, a sort of variables reduction in processes concerning the development/optimisation of ANNs models it is strongly recommended.

6. Conclusions

Artificial Neural Networks, with many alternatives and free of traditional assumptions, are suitable for complex non-linear problems, concerning prediction, modelling or classification. Thus, supervised (e.g. multi-layer feed-forward neural network) and unsupervised (e.g. Kohonen network) algorithms are successfully used in water analysis. ANNs seem to gain attraction during the recent years due to their numerous advantages and become a popular technique for prediction or classification problems.

References

[1] F. Rosenblatt, Psychol. Rev. **65**, 386 (1958).
[2] http://www.generation5.org/content/1999/perceptron.asp (accessed 27 November 2008).
[3] B.G.M. Vandeginste, D.L. Massart, L.M.C. Buydens, S . De Jong, P.J. Lewi, and J. Smeyers-Verbeke, *Handbook of Chemometrics and Qualimetrics: Part B* (Elsevier, Amsterdam, 1998).
[4] B. Kröse and P. Van der Smagt, *An Introduction to Neural Networks* (The University of Amsterdam, Amsterdam, 1996).
[5] Statistica, Version 7.0 (StatSoft Inc., 2004).
[6] D. Svozil, V. Kvasnička, and J. Pospíchal, Chemometr. Intell. Lab. **39**, 43 (1997).
[7] F. Marini, R. Bucci, A.L. Magrì, and A.D. Magrì, Microchem. J. **88**, 178 (2008).
[8] H.T. Nguyen, N.R. Prasad, C.L. Walker, and E.A. Walker, *A First Course in Fuzzy and Neural Control* (Chapman & Hall/CRC, Boca Raton, FL, USA, 2003).
[9] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, Nature **323**, 533 (1986).
[10] M. Kompany-Zareh, A. Massoumi, and Sh. Pezeshk-Zadeh, Talanta **48**, 283 (1999).
[11] V. Sharma, S.C. Negi, R.P. Rudra, and S. Yang, Agr. Water Manage. **63**, 169 (2003).
[12] G.B. Sahoo, C. Ray, and H.F. Wade, Ecol. Model. **183**, 29 (2005).
[13] G. Zhang, Y. Ni, J. Churchill, and S. Kokot, Talanta **70**, 293 (2006).
[14] A.M. Kalteh, P. Hiorth, and R. Berndtsson, Environ. Modell. Softw. **23**, 835 (2008).
[15] F. Marini, Anal. Chim. Acta **635**, 121 (2009).
[16] L. Fausett, *Fundamentals of Neural Networks – Architectures, Algorithms and Applications* (Prentice-Hall, Upper Saddle River, NJ, USA, 1994).
[17] Y.-S. Park, R. Céréghino, A. Compin, and S. Lek, Ecol. Model. **160**, 265 (2003).
[18] T. Kohonen, *Self-organization and Associative Memory*, 3rd ed. (Springer-Verlag, Berlin, Germany, 1989).
[19] J. Zupan, M. Novič, and I. Ruisánchez, Chemometr. Intell. Lab. **38**, 1 (1997).
[20] L.A. Berrueta, R.M. Alonso Salces, and K. Héberger, J. Chromatogr. A **1158**, 196 (2007).
[21] A. Safavi, H. Abdollahi, and M.R. Hormozi Nezhad, Talanta **59**, 515 (2003).
[22] R. Ferrer, J. Guiteras, and J.L. Beltrán, Anal. Chim. Acta **384**, 261 (1999).
[23] H. Elhatip and M.A. Kömür, Environ. Geol. **53**, 1157 (2008).
[24] E.R. Rene and M.B. Saidutta, Int. J. Environ. Res. **2**, 183 (2008).
[25] H. Tutu, E.M. Cukrowska, V. Dohnal, and J. Havel, Environ. Model. Assess. **10**, 143 (2005).
[26] B. Purkait, S.S. Kadam, and S.K. Das, J. Environ. Inform. **12**, 140 (2008).
[27] S.W. Chung and J.H. Kim, Water Sci. Technol. **52**, 83 (2005).
[28] M.I. Yesilnacar, E. Sahinkaya, M. Naz, and B. Ozkaya, Environ. Geol. **56**, 19 (2008).
[29] J.P. Suen and J.W. Eheart, J. Water Resour. Plng. Mgmt. **129**, 505 (2003).
[30] A. Gemitzi, C. Petalas, V. Pisinaras, and V.A. Tsihrintzis, Hydrol. Process. **23**, 372 (2009).
[31] E. Dogan, B. Sengorur, and R. Koklu, J. Environ. Manage. **90**, 1229 (2009).
[32] B. Sengorur, E. Dogan, R. Koklu, and A. Samandar, Fresen. Environ. Bull. **15**, 1064 (2006).
[33] W. Huang and S. Foo, Water Res. **36**, 356 (2002).
[34] J.F. Fernández-Sánchez, A.S. Carretero, J.M. Benítez-Sánchez, C. Cruses-Balnco, and A. Fernández-Gutiérrez, Anal. Chim. Acta **510**, 183 (2004).
[35] G. Sahoo, C. Ray, E. Mehnert, and D.A. Keefer, Sci. Total Environ. **367**, 234 (2006).
[36] Y. Ni, Q. Ping, and S. Kokot, Anal. Chim. Acta **537**, 321 (2005).
[37] D. Brodnjak-Vončina, D. Dobčnik, M. Novič, and J. Zupan, Anal. Chim. Acta **462**, 87 (2002).
[38] A. Astel, S. Tsakovski, P. Barbieri, and V. Simeonov, Water Res. **41**, 4566 (2007).
[39] A. Astel, S. Tsakovski, V. Simeonov, E. Reisenhofer, S. Piselli, and P. Barbieri, Anal. Bioanal. Chem. **390**, 1283 (2008).
[40] T. Kuzniz, D. Halot, A.G. Mignani, L. Ciaccheri, K. Kalli, M. Tur, A. Othonos, C. Christofides, and D.A. Jackson, Sensor. Actuat. B-Chem. **121**, 231 (2007).
[41] J.C. Chen, N.B. Chang, and W.K. Shieh, Eng. Appl. Artif. Intel. **16**, 149 (2003).
[42] Ö. Çinar and H. Merdun, Ecol. Res. **24**, 163 (2009).

[43] S. Palani, S.-Y. Liong, and P. Tkalich, Mar. Pollut. Bull. **56**, 1586 (2008).
[44] D.B. May and M. Sivakumar, Environ. Modell. Softw. **24**, 296 (2009).
[45] J.-T. Kuo, M.-H. Hsieh, W.-S. Lung, and N. She, Ecol. Model. **200**, 171 (2007).
[46] N. Grošelj, G. Veer, M. Tušar, M. Vračko, and M. Novič, Food Chem. **118**, 941 (2010).
[47] W. Melssen, R. Wehrens, and L. Buydens, Chemometr. Intell. Lab. **83**, 99 (2006).
[48] A. Ramil, A.J. López, and A. Yáñez, Appl. Phys. A **92**, 197 (2008).
[49] M. Wesolowski, B. Suchacz, and J. Halkiewicz, Anal. Bioanal. Chem. **384**, 458 (2006).
[50] R.A. Verdini, S.E. Zorrrilla, A.C. Rubiolo, and S. Nakai, Chemometr. Intell. Lab. **86**, 60 (2007).