A simple neural network model for the determination of aquifer parameters

N. Samani a,*, M. Gohari-Moghadam a, A.A. Safavi b

a Department of Earth Sciences, College of Sciences, Shiraz University, Shiraz 71454, Iran
b School of Engineering, Shiraz University, Iran

Received 30 April 2006; received in revised form 19 March 2007; accepted 22 March 2007

Summary A simple artificial neural network (ANN) model is developed for the determination of non-leaky confined aquifer parameters by normalizing and applying the principal component analysis (PCA) on adopted training data pattern from Lin and Chen [Lin, G.F., Chen, G.R., 2006. An improved neural network approach to the determination of aquifer parameters. Journal of Hydrology 316 (1–4), 281–289]. The proposed network uses faster Levenberg–Marquardt training algorithm instead of gradient descent. The application of PCA highly reduced the network topology so that it has only one neuron in the input layer and eight neurons in the hidden layer regardless of the number of drawdown records in the pumping test data. The network trained with 10,205 training sets and tested with 2000 sets of synthetic data. The network generates the coordinates of the match point for any individual pumping test case study and then the aquifer parameters are calculated using Theis’ equation. The simple ANN trains faster and determines the coordinate of the match point more accurately because of the simplified topology and LM training algorithm. The accuracy, generalization ability and reliability of the proposed network is verified by two sets of real-time field data and the results are compared with that of Lin and Chen as well as graphical methods of aquifer parameters estimation. The proposed ANN appears to be a simpler and more accurate alternative to the type curve-matching techniques and previous ANN methods.

Introduction

The Theis non-equilibrium solution (1935) is commonly used to calculate the transmissivity ($T$) and storage coefficient ($S$) of infinite homogeneous, isotropic confined aquifer from constant rate pumping test data. According to Theis’ method for the drawdown $s$, the exact solution is...
\[ s = \frac{Q}{4\pi T} W(u), \]

where \( T \) is transmissivity (m²/day), \( Q \) is constant discharge rate of pumped well (m³/day), and \( W(u) \) is known as the Theis well function. The well function is an exponential integral

\[ W(u) = \int_{u}^{\infty} \frac{e^{-u}}{u} \, du, \]

in which \( e \) is the base of natural logarithm and \( u \) is an argument known as dimensionless time expressed as

\[ u = \frac{r^2 S}{4Tt}, \]

where \( t \) is pumping time (day) and \( r \) is distance from observation point to pumping well (m).

For practical applications, Theis suggested a graphical type-curve matching technique for the solution of \( S \) and \( T \) from Eqs. (1)–(3). Using this method, values of drawdown \( s \) are plotted against values of \( t \) on logarithmic paper to the same scale as the "type curve" resulting in a curve of observed data. The type curve is a curve showing the relation between the argument \( u \) and the well function \( W(u) \). The curve of the observed data are then superimposed on the type curve in such a way that the coordinate axes of the two curves are kept in parallel. A position is found by trial, whereby most of the plotted points of the observed data fall on a segment of the type curve. Any convenient point is selected, and the coordinates of this match point are recorded. With values of \( W(u), u, t \) and \( s \) thus determined, \( S \) and \( T \) can be obtained. The aquifer parameters obtained by the type-curve graphical method are of uncertain reliability because of personnel matching errors. In recent years, some convenient and more precise approaches based on artificial neural networks (ANNs) have been developed (for example, Aziz and Wong, 1992; Zio, 1997; Balkhair, 2002; Garcia and Shigidi, 2006; and very recently Lin and Chen, 2005).

Artificial neural networks, which originated about 60 years ago (McCulloch and Pitts, 1943), were inspired by a desire to emulate human learning. Mathematically, an ANN is a universal approximator, proven to be highly effective for modeling non-linear problems, with application to a diversity of large-scale problems, including pattern recognition, classification, and control. An ANN is arranged into discrete layers, consisting of input, hidden, and output, each of which includes one or more individual nodes or processing elements. The number of input variables necessary for predicting the desired output variable determines the number of input nodes. The optimum number of hidden nodes and hidden layers is dependent on the complexity of the modeling problem and the objectives of the researcher, such as the acceptable learning error (Faussett, 1994). During training, the ANN is presented with patterns of input and corresponding output pairs, during which the learning algorithm iteratively adjusts the values of connection weights within the ANN structure. In general, it is more desirable to attain the desired accuracy with a simpler ANN structure (i.e., fewer nodes), as this can reduce training time, improve network generalization and prevent over-fitting (Hagan et al., 1996). Rumelhart et al. (1986) proposed a supervised learning approach for neural networks known as back-propagation training algorithm. In this training process, the initial connection weights of nodes in the network change to minimize the overall network error for all input patterns in the training set.

The development of more powerful computers in terms of processing speed and random access memory makes the ANN solution of complex systems possible. Another factor that motivates an interest in ANNs is the increasing availability of large data sets from continuous data loggers/data collection systems, such as SCADA (Supervisory Control and Data Acquisition). The ANNs have been used as an alternative modeling tool for certain hydrologic and hydrogeologic problems. The use of ANN models in water resources has grown considerably over the last decade (for example, Rizzo and Dougherty, 1994; Rogers and Dowla, 1994; Hsu et al., 1995; Clair and Ehrman, 1996; Maier and Dandy, 1996; Poff et al., 1996; Shamseldin, 1997; Gumrah et al., 2000; Coulbaly et al., 2001; Lin and Chen, 2004a,b; Lallahem et al., 2005).

In recent years, the ANN has been used as an alternative approach for the estimation of aquifer parameters, although the early articles have some technical problems. Very recently, Lin and Chen (2006) proposed a procedure to determine confined aquifer parameters based on the combination of a new ANN approach and Theis (1935) analytical solution (Eqs. (1)–(3)) and has some advantages when compared with previous ANN approaches. It avoids the problem regarding the selection of a suitable trained range of aquifer parameters. Lin and Chen approach which is referred to as Lin–Chen network has a topology of \((N - 1) \times 32 \times 1\), where \(N - 1\), 32 and 1 are the number of input (\(N\) is the number of drawdown measurements), hidden and output layer neurons, respectively. The drawback of this approach is that as the number of time-drawdown measurements (\(N\)) increases, such as data in the continuously recording data loggers, the dimensionality of input layer is also increased. Consequently, the error surface is more complicated and the application of original Lin–Chen ANN is hardly possible.

In this paper, first we modified the original Lin–Chen network by replacing the gradient descent algorithm with Levenberg–Marquardt (LM) training algorithm, which helped reduce the convergence criterion from \(10^{-3}\) to \(10^{-9}\). Then, a simple ANN is proposed by normalizing the Lin and Chen training pattern and applying principal component analysis (PCA). PCA detects the structure in the relationships between input variables in the training data set by explaining the correlation amongst training patterns in terms of a smaller number of underlying factors (principal components or PCs) by eliminating redundant data (Jackson, 1991; Meglen, 1992). Consequently, the dimension of input layer is reduced from \(N - 1\) to only one. This further reduces the number of hidden layer neurons from 32 to 8. As a result, the combination of LM algorithm and PCA not only improves the prediction performance and enhances the training speed but it also reduces the topology of the proposed ANN to \(1 \times 8 \times 1\), regardless of the number of drawdown measurements.

The proposed ANN has the following advantages over the original Lin–Chen networks. First, it uses the high performance LM training algorithm that causes faster training...
and higher accuracy. Second, it has a highly reduced topology that simplifies the network training process, further increases parameter estimation accuracy and reduces the training time by a factor of 10 or more. This simplification is very valuable for analyzing many alternative pumping tests routinely obtained from continuous data loggers/data collection systems. Third, its topology is fixed and independent of number of time-drawdown measurements in the pumping test data. Finally, the proposed simple ANN appears to be an alternative to the existing ANNs and traditional type curve matching techniques.

Methodology

Feed-forward neural network (FNN)

Artificial neural networks are computing tools constructed of many simple interconnected elements called neurons with a unique capability of recognizing underlying relationships between input and output events. Fig. 1 shows a typical neuron (Dayhoff, 1990). A neuron has two components (1) a weighted sum \( s = \sum W_i x_i + b \) that performs a weighted summation of the inputs \( x_1, x_2, x_3, \ldots x_n \), where \( b \) is the bias of the network and (2) a linear, nonlinear or logic transfer function which gives an output corresponding to \( s \). Here, many kinds of functions, including threshold (logic), sigmoid, hyperbolic tangent, Gaussian and linear could be used. In this paper, hyperbolic tangent (tansig) function \( f(x) = \frac{1}{1 + \exp(-x)} \) is applied for input and hidden layers (Fig. 2). In addition, a linear transfer function (purelin) applied for output layer (Fig. 2).

In a typical ANN, there are three types of neurons (Hsu et al., 1995): (a) input neurons that may receive external data, (b) output neurons that send data out of the ANN, and (c) hidden neurons whose signals remain within the ANN and connect the input layer neurons to output layer neurons (Fig. 3). Therefore, there are three types of layers corresponding to the types of neurons. The hidden neurons may form one or more hidden layers. The neurons in each layer are usually fully interconnected with neurons from neighboring layers. The importance of each inter-neuron connection is determined by its numerical value that is named weights. A three-layered back-propagation network structure is depicted in Fig. 3 (Dayhoff, 1990; Hsu et al., 1995).

The ANN shown in Fig. 3 has an input layer, an output layer, and one hidden layer. The input layer contains an array of variables into which the input data of the system are read from an external source. Similarly, the predicted data or results, which can be multiple vectors, are written in the output layer. Initially, the input layer receives the input and passes it to the hidden layer. If more hidden layers exist, the processed information from the first hidden layer is then...
passed to the next hidden layer for processing. Finally, the output layer receives information from the last hidden layer. In feed forward networks the flow of computation is one sided from input layer to output layer. In this paper, a three-layered ANN with a simplified structure is designed.

When an ANN is constructed, small numbers (weights) are assigned randomly to the connections between neurons (initialization). In general, the output from neuron $j$ in layer $k$ can be calculated by the following equation:

$$u_{jk} = F_k \left( \sum_{i=1}^{N_{in}} w_{ijk} u_{i(k-1)} + b_{jk} \right),$$  

(4)

where coefficients $w_{ijk}$ and $b_{jk}$ are connection weight and bias of the network, respectively; they are fitting parameters of the model, $N_{in}$ is the total number of inputs applied to neuron $i$ in layer $k-1$. The purpose is to obtain a mapping from an input vector to an output one. It is desired that the difference between the predicted and the observed (actual or synthetic) values in the output vector be as small as possible. The fitting parameters are modified until an error criterion between the input and the output is satisfied based on the topology of the ANN and the learning technique. The adjustment of the weights and biases of each neuron is the learning or training process that in comparison is equal to parameter calibration in conventional water resources modeling.

Lin and Chen approach

Lin and Chen (2006), suggested a training method for looking for the best match point. They argue that in the type curves presented by Walton (1962), log(1/u) is always greater than –0.5. While log (1/u) is greater than 4.0, log W(u) approaches a constant. Hence, the trained range of their ANN output, log(1/u_m), is selected from –0.5 to 4.0. Once a specific log(1/u_m) is chosen, the W(u_m) value is calculated. Then, the input vector components for training are generated as

$$x_i = \log \left[ \frac{W(u_m t_{im} / t_{1m})}{W(u_m)} \right], \quad i = 1, 2, \ldots, N - 1.$$  

(5)

The Lin–Chen ANN is designed to produce the 1/u coordinate of the match point (1/u_m) as

$$y = \log \left[ \frac{1}{u_m} \right].$$  

(6)

Lin and Chen (2006) employed the gradient descent backpropagation algorithm with momentum and adaptive learning rate (GDX) to train their ANN. The change of weights is

$$\Delta w^{k-1} (n+1) = \eta \delta^1 \eta^{k-1} + z \Delta w^{k-1} (n),$$  

(7)

where $n$ is the iteration number, $\delta^1$ is the output from neuron $i$ in layer $k-1$, $\eta$ is the learning-rate parameter and $z$ is the momentum constant. The $\delta^k$ can be calculated by

$$\delta^k = \begin{cases} e_j \frac{\partial f}{\partial y_j} & \text{for neuron } j \text{ in output layer}, \\ \frac{\partial f}{\partial y_j} \sum_{i=1}^{N_{out}} \delta^l w^{k+1}_{ji} & \text{for neuron } j \text{ in hidden layer } k, \end{cases}$$  

(8)

where $f$ is the output function for each neuron, $m_{k+1}$ is the total number of inputs applied to neuron $r$ in layer $k + 1$, and $e_j$ is the error at the output of neuron $j$. The training process is an iterative process for determining appropriate weights that minimize the objective function $F$:

$$F = \frac{1}{2} \sum_{i=1}^{N_{train}} \sum_{j=1}^{N_{out}} (d_{ij} - y_{ij})^2,$$  

(9)

where $d_{ij}$ is the desired output, $y_{ij}$ is the actual output, $N_{train}$ is the number of patterns in the training data set and $N_{out}$ is the number of neurons in the output layer. Learning continues until $F$ converges to a convergence criterion, which is an acceptably small value. The relative root mean square error (RMSE) of training process is 1% for the Lin–Chen network.

After training, it is capable of producing an output with desired accuracy, when a transformed time-drawdown data vector employed as input. To determine the aquifer parameters from a set of $N$ observed time-drawdown data, Lin–Chen network has $N - 1, 32$ and $1$ neurons in input, hidden and output layers, respectively (note that number of neurons in the hidden layer is defined by trial and error for the best network performance). The components of the input vector are obtained from $N$ observed time-drawdown data as

$$x_i = \log(s_{i+1}) - \log(s_i) = \log \left( \frac{S_{i+1}}{S_i} \right), \quad i = 1, 2, \ldots, N - 1,$$  

(10)

where $s_i$ is the drawdown observed at time $t_i$. Then the coordinates of the match point $W_m$, $s_m$, $1/u_m$ and $t_m$ are determined from the output and time-drawdown data as follows:

$$\frac{1}{u_m} = 10^y,$$  

(11)

$$W_m = W(u_m) = W \left( \frac{1}{10^y} \right).$$  

(12)

$$s_m = s_i,$$  

(13)

and

$$t_m = t_i.$$  

(14)

Then by applying the Theis (1935) analytical solution transmissivity $T$ and storage coefficient $S$ is determined by

$$T = \frac{Q W_m(u)}{4\pi s_m},$$  

(15)

and

$$S = \frac{4 T u_m t_m}{r^2},$$  

(16)

where $W_m$, $s_m$, $u_m$ and $t_m$ are the coordinates of the match point.

Levenberg–Marquardt training algorithm

In this paper, the backpropagation technique with LM method is used to train the ANNs. In general, the back-propaga-
tion method uses the following steps. First, reads a specific input and calculates its corresponding output. Second, if the error between the produced output and the desired output is acceptable, then stops. Third, if the error is unacceptable in the previous step, then the weights are adjusted for all the interconnections that go into the output layer. An error value is calculated for all the units in the hidden layer that is just below the output layer, and the weights are then adjusted for all interconnections that go into the hidden layer. The process continues until the last layer of weights adjusts. Typically, an application of back-propagation requires both a training set and a test set. Both sets contain input/output pattern pairs (Dayhoff, 1990).

While the training set is used to train the network, the test set is used to assess the performance of the network after the training is complete. To provide the best test of network performance, the test set should be different from the training set. The most successful ANN architecture is the one that has the smallest prediction error on the data set for which it was not trained. In addition, the error criterion for the network must be considered carefully to avoid the problem of over training and over fitting in the training phase. Therefore, special attention should be applied in the ANN design to generalize the relationship in training phase. Therefore, in this article a three-layered feed forward neural network with back propagation LM training algorithm is applied to construct a network to evaluate the match point coordinate for a non-leafy confined aquifer.

Principal component analysis

Sometimes, the dimension of the input vector is large, but the components of the vectors are highly correlated (redundant). It is useful in this situation to reduce the dimension of the input vectors. PCA is an effective procedure for reducing the dimensionality of large data sets. It permits identification of associations between variables, therefore reducing the dimensionality of the data set. (Wold et al., 1987; Jolliffe, 1986; Everitt and Dunn, 1992).

This technique has three effects: (a) it orthogonalizes the components of the input vectors, so that it produces the uncorrelated orthogonal variables or PCs by multiplying the original correlated variables with the eigenvector. (b) It orders the resulting orthogonal components (principal components) so that those with the largest variation come first. (c) It eliminates those components that contribute the least to the variation in the data set.

The input vectors should be normalized, so that they have zero mean and unit variance. This is a standard procedure when using principal components. One of the parameters in PCA method is minimum fraction variance. It determines the elimination of those principal components that contribute less than this value to the total variation in the data set. For example, if the minimum fraction variance is considered 0.02 then PCA eliminates those principal components that contribute less than 2% to the total variation in the data set.

The principal components are described by two lower dimensional data matrices (scores and loadings) that describe the underlying patterns within the original data (Wold et al., 1987). The scores matrix is the data formed by transforming the original data into the space of the principal components. Rows of score correspond to observations and columns to components. The Eigen analysis of a $p \times p$ correlation matrix produces $p$ pairs of eigenvalues and eigenvectors. Each eigenvalue/eigenvector pair describes a principal component. The eigenvalues describe the amount of variance explained by each principal component and the loadings are the coordinates of the eigenvector. By ordering the eigenvectors in the order of descending eigenvalues (largest first), one can create an ordered orthogonal basis with the first eigenvector having the direction of largest variance of the data. In this way, we can find directions in which the data set has the most significant amounts of energy. The principal component scores are then given as linear combinations of the original standardized (auto scaled) data with the loadings as the coefficients. Principal components are extracted so that the maximum amount of variance is explained in (has the largest eigenvalue associated with) the first principal component and progressively less variance is explained for each subsequent component.

Consequently, the PCA technique extracts the eigenvalues of the PCs that are the measure of variance in the observations and eigenvectors (loadings or weightings) which determines the participation of the original variables in the PCs from the covariance matrix of original variables. Suppose we have a random vector population $x$, where

$$ x = (x_1, \ldots, x_n)^T $$

and the mean of that population is denoted by
and the covariance matrix of the same data set is
\[
C_x = E\{ (x - \mu_x)(x - \mu_x)^T \}. \tag{20}
\]

The components of \( C_x \), denoted by \( c_{ij} \), represent the covariance between the random variable components \( x_i \) and \( x_j \). The component \( c_{ii} \) is the variance of the \( x_i \) component. The variance of a component indicates the spread of the component values around its mean value. If two components \( x_i \) and \( x_j \) of the data are uncorrelated, their covariance is zero (\( c_{ij} = c_{ji} = 0 \)). The covariance matrix is, by definition, always symmetric.

From a symmetric matrix such as the covariance matrix, we can calculate an orthogonal basis by finding its Eigenvalues and Eigenvectors. The eigenvectors \( \mathbf{e}_i \) and the corresponding eigenvalues \( \lambda_i \) are the solutions of the equation
\[
C_x \mathbf{e}_i = \lambda_i \mathbf{e}_i, \quad i = 1, \ldots, n. \tag{21}
\]
These values can be found, for example, by finding the solutions of the characteristic equation
\[
|C_x - \lambda I| = 0 \tag{22}
\]
where \( I \) is the identity matrix having the same order than \( C_x \) and the \( | \cdot | \) denotes the determinant of the matrix. If the data vector has \( n \) components, the characteristic equation becomes of order \( n \). This is easy to solve only if \( n \) is small. Solving eigenvalues and corresponding eigenvectors is a non-trivial task, and many methods exist.

In this paper, the Principal components analysis of the ANN input data sets were performed using MATLAB (2004). The multivariate statistical technique of principal component analysis PCA has recently been applied to environmental assessment (Zitko, 1994). Haan (1977) gives a good overview of the method with applications in hydrology.

Fig. 4 shows the structure of Lin–Chen and the proposed ANNs. The proposed ANN not only avoids the problem concerning the selection of an appropriate trained range based on the Lin and Chen (2006) training pattern, but its dimensionality of the input training set is also reduced from \( N = 1 \) to 1 due to application of PCA on the original data. In addition, due to lower dimension of input data, it is trained with only eight neurons in the hidden layer (unlike Lin–Chen network which has 32 neurons in the hidden layer). Due to simpler structure, our proposed network is trained more rapidly than the Lin–Chen network and it has greater accuracy. These advantages will be further verified in the following section.

Our ANN composed of three layers that are designed to produce values of \( \log(1/\bar{u}_n) \) when the first principal component (PCA) of the proposed training set is provided as input. It includes one, eight and one neurons in the input, hidden and output layers, respectively. The components of the input and output vectors are shown in Fig. 4.

Therefore, the major difference between our network and the Lin-Chen network is the design of input components, input and hidden layers structures (i.e., fewer nodes) and its training algorithm, which is LM back-propagation training algorithm.

Criteria of evaluation

The performance of a trained network is measured to some extent by the errors on the validation and test sets. One operation is to perform a linear regression analysis between the network response and the corresponding targets. If we had a perfect fit (outputs exactly equal to targets), the slope would be 1, and the \( y \)-intercept would be zero.

However, the correlation coefficient (\( R \)) between the outputs and targets is a measure of how well the variation in the target is explained by the outputs. If this number is equal to 1, then there is a perfect correlation between targets and outputs (Fig. 5).

Here, two different criteria are used to evaluate the effectiveness of each network and its ability to make precise predictions. The RRMSE of the estimated drawdown given by
\[
\text{RRMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{n} \left( \frac{y_i - \hat{y}_i}{y_i} \right)^2}, \tag{23}
\]
where \( y_i \) is the simulated values computed by the network, \( \hat{y}_i \) is the target values and \( n \) is the number of data points. The lowest the RRMSE, the more accurate the prediction is. Another criterion is efficiency \( R^2 \), given by
\[
R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum y_i^2 - \overline{y}^2}, \tag{24}
\]
represents the percentage of the initial uncertainty explained by the model. The best fit between observed and calculated values, which is unlikely to occur, would have RRMSE = 0 and \( R^2 = 1 \).
Application and discussion

In this section, the proposed and Lin–Chen ANNs have trained individually with 10,205 synthetic training sets using LM algorithm. Lin–Chen ANN trained by \( N - 1 \) input neurons, 32 hidden neurons and one output neuron while, in the proposed ANN, before training, the data set is normalized to a new set that have zero mean and unit variance and after that, the principal components of the new set are extracted by considering minimum fraction variance of 0.001. Therefore, it has only one input neuron, eight hidden layer neurons (due to lower dimensionality of input neurons) and one output neuron. Then, the accuracy of both trained ANNs was tested with 2000 sets of synthetic data that is not seen before by the networks in the training process. The synthetic data sets were constructed for a predefined \( T \) and \( S \) values that converted to drawdown ratios by Theis equation (Eqs. (1)–(3)), and Lin and Chen (2006) approach. The results are compared according to RMSE error criterion and regression parameters between idealized and estimated (network output) parameter values.

Further, two real time-drawdown data sets that are adapted from Todd and Mays (2005), and Kruseman and de Ridder (1992) have been tested the applicability and reliability of both ANNs. The results are also compared with graphical methods.

Example 1. training and testing the proposed and Lin–Chen ANNs using synthetic data.

The total number of 10,205 training patterns are synthetically generated such that the dimensionless drawdown ratios is considered as input and \( \log(1/u_n) \) values ranging from −0.5 to 4.0 with a step size of \( 4.41 \times 10^{-3} \) as target (using Eqs. (5), (6) and Theis solution). Both the proposed and Lin–Chen networks have three layers and one neuron in the output layer. Since there are 25 time steps in Todd and Mays (2005) pumping test example, therefore, Lin–Chen ANN has 24 neurons in its input layer. The proposed ANN has always only one neuron in its input layer regardless of number of time steps in time-drawdown data, because the inputs training set is normalized to a new data set with zero mean and unit standard deviation and the principal components of the standardized set is computed. Then the first PC that account for 99.911% of the variance in the data set is considered as input. The amount of variance (i.e., information) spanned by each PC depends on the relative value of its eigenvalue with respect to the total sum of eigenvalues (Jackson, 1991). The amount of eigenvalue, variance and cumulative variance for three first PCs of normalized input training sets are presented in Table 1. As can be seen, the highest data variation occurs in the first principal component. However, since the dimension of input data have been reduced to a one-dimensional vector, it was possible to reduce the number of hidden layer neurons from 32 in the Lin–Chen ANN to 8 neurons in our proposed network.

The proposed and Lin–Chen ANNs are trained with the same key parameters and the same number of training patterns. The ANNs significant parameters used during training are presented in Table 2.

Table 3 summarizes the ANN structures and the required training times for the proposed and the modified Lin–Chen ANNs. The required training time for the modified Lin–Chen and the proposed ANNs are 1794.422 and 166.375 seconds on an Intel 2.8 GHz personal computer with 1 GB RAM, respectively. Therefore, the proposed ANN is trained 10 times faster than the modified Lin–Chen ANN. This is due to its simpler structure and its far lower number of neurons in its input and hidden layers. The RMSE values for the proposed and the modified Lin–Chen ANNs during the training process has been summarized in Table 4. The RMSE values for the proposed and the modified Lin–Chen ANNs trained by LM algorithm are 0.221% and 0.165%, respectively. Therefore, using the LM training algorithm improve the performance of networks as compared to the original Lin–Chen (GDX) network whose RMSE value is 1%. Table 5 shows the regression parameters between output and target values for

| Table 1 | The principal component parameters of the training set |
|----------------|------------------|------------------|------------------|
| Principal component | Eigenvalue | Variance (%) | Cumulative variance (%) |
| PC1 | 10.287 | 99.911 | 99.911 |
| PC2 | 9.2e−3 | 0.089 | 100 |
| PC3 | 1.0453e−5 | 0 | 100 |

| Table 2 | The ANN parameters applied during training |
|----------------|------------------|------------------|
| Parameter | Value |
| Learning rate | 0.5 |
| Momentum constant | 0.6 |
| Convergence criterion | 1e−9 |
| Maximum training cycle | 10,000 |
| Number of training patterns | 10,205 |
the proposed and the modified Lin–Chen ANNs. As it is seen, the accuracy of both networks is very high.

One of the problems that occur during neural network training is called over fitting. The error on the training set is driven to a very small value, but when new data are presented to the network the error is large. The network has memorized the training examples, but it has not learned to generalize to new situations. To evaluate the generalization of the proposed and the modified Lin–Chen ANNs, 2000 test patterns that are not used during the training process are employed. The tested patterns are generated from combinations of idealized $T$ and $S$ values ranging from $10^2$ to $10^6$ m$^2$/day with 500 step size and $10^{-2}$ to $10^{-9}$, with $5 \times 10^{-6}$ step size, respectively. Figs. 6 and 7 show the scatter plots and the best-fitted line for the idealized and estimated aquifer parameters ($T$ and $S$) obtained by the proposed and the modified Lin–Chen ANNs. As shown in these figures, both networks are capable of accurately estimating aquifer parameters over a wide tested range. The RRMSE values for the proposed and the modified Lin–Chen ANNs during the training process are summarized in Table 6. As a result, the proposed network can better generalize the data pattern and so it has lower RRMSE values for both ranges of the $T$ and $S$ values. However, the proposed ANN is a simpler and faster as compared to the Lin–Chen ANN. Both ANNs with LM training algorithm are more accurate and more rapidly trained than the original Lin–Chen network that uses GDX training algorithm. Moreover, there is less of a tendency for the search to become trapped in local minima, and consequently the networks are trained to a lower performance criterion ($1 \times 10^{-9}$).

Furthermore, the proposed and the modified Lin–Chen ANNs with LM training algorithm that trained with synthetic time-drawdown data have also been tested with two sets of field data. The results have been compared with each other and with graphical and mathematical matching procedures. The first set of data adapted from a test presented in Todd and Mays (2005). The second set of data are for the “Oude Korendijk” site adapted from Kruseman and de Ridder (1992), Table 7.

### Table 3  Number of neurons and the required training times for the proposed and Lin–Chen ANNs

<table>
<thead>
<tr>
<th>ANN</th>
<th>Number of neurons</th>
<th>Required training time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Input layer</td>
<td>Hidden layer</td>
</tr>
<tr>
<td>Modified Lin–Chen by LM</td>
<td>24</td>
<td>32</td>
</tr>
<tr>
<td>Proposed (LM, Prestd. &amp; PCA)</td>
<td>1</td>
<td>8</td>
</tr>
</tbody>
</table>

LM, Levenberg–Marquardt; Prestd., pre-normalization; PCA, principal component analysis.

### Table 4  RRMSE values for the proposed and Lin–Chen ANNs during the training process

<table>
<thead>
<tr>
<th>ANN</th>
<th>RRMSE (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lin–Chen with (GDX)</td>
<td>1</td>
</tr>
<tr>
<td>Modified Lin–Chen by (LM)</td>
<td>0.165</td>
</tr>
<tr>
<td>Proposed (LM, Prestd. &amp; PCA)</td>
<td>0.221</td>
</tr>
</tbody>
</table>

GDX, gradient descent; LM, Levenberg–Marquardt.

### Table 5  Regression parameters for the proposed and the modified Lin–Chen ANNs during the training process

<table>
<thead>
<tr>
<th>ANN</th>
<th>Slope</th>
<th>Intercept</th>
<th>$R^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modified Lin–Chen (LM)</td>
<td>1</td>
<td>-5.64e-7</td>
<td>1</td>
</tr>
<tr>
<td>Proposed (LM, Prestd., &amp; PCA)</td>
<td>1</td>
<td>-3.96e-7</td>
<td>1</td>
</tr>
</tbody>
</table>

Idealized $T$ (m$^2$/day) 

**Figure 6** Idealized $T$ values versus estimated $T$ values obtained by (a) the modified Lin–Chen and (b) the proposed ANNs.
Example 2. Testing the proposed and modified Lin–Chen ANNs using field data from Todd and Mays (2005). The data of this example are taken from Todd and Mays (2005). In this example, a well that fully penetrating a confined aquifer was discharged with a uniform pumping rate of 2500 m³/day. The time-drawdown data were measured in a piezometer at 60 m from the pumping well (Table 7). The trained proposed and modified Lin–Chen ANNs have been verified and compared with these data results in T and S values that are represented in Table 8. Here, the modified Lin–Chen ANN dimension is $24 \times 32 \times 1$ according to 25 observed time steps, but the proposed ANN dimension is $1 \times 8 \times 1$. As it is observed, the ANNs results are in good agreement with other published results. Therefore, both networks may be used as an alternative for conventional type curve matching methods with the preference of the proposed network.

Example 3. Testing the proposed and Lin–Chen ANNs using field data from Kruseman and de Ridder (1992). The data of this example are taken from Kruseman and de Ridder (1992). In this Example, a fully penetrating well discharges a confined aquifer with a uniform pumping rate of 788 m³/day. The time-drawdown data are measured in a piezometer at 90 m from the pumping well, Table 7. Note that these data are affected by systematic noise due to pumping rate variations, interference effects, etc. (Fig. 3.6 in p. 64 of Kruseman and de Ridder, 1992). The trained proposed and modified Lin–Chen ANNs have been verified with these data comparing the resulting T and S values.

### Table 7: Reported time–drawdown data of examples 2 and 3

<table>
<thead>
<tr>
<th>Example</th>
<th>Time (min)</th>
<th>Drawdown (m)</th>
<th>Time (min)</th>
<th>Drawdown (m)</th>
<th>Time (min)</th>
<th>Drawdown (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example 2 (Todd and Mays, 2005)</td>
<td>1.0</td>
<td>0.2</td>
<td>1.5</td>
<td>0.015</td>
<td>180</td>
<td>0.569</td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>0.27</td>
<td>2.0</td>
<td>0.021</td>
<td>248</td>
<td>0.593</td>
</tr>
<tr>
<td></td>
<td>2.0</td>
<td>0.30</td>
<td>2.16</td>
<td>0.023</td>
<td>301</td>
<td>0.614</td>
</tr>
<tr>
<td></td>
<td>2.5</td>
<td>0.34</td>
<td>2.66</td>
<td>0.044</td>
<td>363</td>
<td>0.636</td>
</tr>
<tr>
<td></td>
<td>3.0</td>
<td>0.37</td>
<td>3.0</td>
<td>0.054</td>
<td>422</td>
<td>0.657</td>
</tr>
<tr>
<td></td>
<td>4.0</td>
<td>0.41</td>
<td>3.5</td>
<td>0.075</td>
<td>452</td>
<td>0.679</td>
</tr>
<tr>
<td></td>
<td>5.0</td>
<td>0.45</td>
<td>4.0</td>
<td>0.090</td>
<td>602</td>
<td>0.688</td>
</tr>
<tr>
<td></td>
<td>6.0</td>
<td>0.48</td>
<td>4.33</td>
<td>0.104</td>
<td>680</td>
<td>0.701</td>
</tr>
<tr>
<td></td>
<td>8.0</td>
<td>0.53</td>
<td>5.5</td>
<td>0.133</td>
<td>785</td>
<td>0.718</td>
</tr>
<tr>
<td></td>
<td>10.0</td>
<td>0.57</td>
<td>6.0</td>
<td>0.153</td>
<td>845</td>
<td>0.716</td>
</tr>
<tr>
<td></td>
<td>12.0</td>
<td>0.60</td>
<td>7.5</td>
<td>0.178</td>
<td>14.0</td>
<td>0.206</td>
</tr>
<tr>
<td></td>
<td>14.0</td>
<td>0.63</td>
<td>9.0</td>
<td>0.220</td>
<td>18.0</td>
<td>0.250</td>
</tr>
<tr>
<td></td>
<td>24.0</td>
<td>0.72</td>
<td>15.0</td>
<td>0.275</td>
<td>30.0</td>
<td>0.305</td>
</tr>
<tr>
<td></td>
<td>40.0</td>
<td>0.81</td>
<td>25.0</td>
<td>0.348</td>
<td>50.0</td>
<td>0.364</td>
</tr>
<tr>
<td></td>
<td>60.0</td>
<td>0.90</td>
<td>40.0</td>
<td>0.404</td>
<td>80.0</td>
<td>0.429</td>
</tr>
<tr>
<td></td>
<td>100.0</td>
<td>0.96</td>
<td>60.0</td>
<td>0.444</td>
<td>120.0</td>
<td>0.476</td>
</tr>
<tr>
<td></td>
<td>150.0</td>
<td>1.04</td>
<td>90.0</td>
<td>0.494</td>
<td>180.0</td>
<td>0.507</td>
</tr>
<tr>
<td></td>
<td>210.0</td>
<td>1.10</td>
<td>120.0</td>
<td>0.528</td>
<td>240.0</td>
<td>0.550</td>
</tr>
</tbody>
</table>

Figure 7: Idealized S values versus estimated S values obtained by (a) the modified Lin–Chen and (b) the proposed ANNs.
values that are represented in Table 9. In this case, the modified Lin–Chen ANN dimension is $34 \times 32 \times 1$ according to 35 observed time steps but the proposed ANN dimension is $1 \times 8 \times 1$. Consequently, the training time for the first ANN was 889.734 s and for the simplified ANN is 164.922 s. Therefore, the proposed network yields a training time reduction of 82% as compared to the modified Lin–Chen network. This example demonstrates the capability of the proposed network as a simple and reliable substitution for conventional graphical matching techniques in determining the aquifer parameters from noisy pumping test data.

Summary and conclusion

In recent years, ANNs have found many applications in various aspects of hydrology including aquifer parameters determination from pumping test data. Because of the incognito of natural range of aquifer parameters for an unknown reservoir, the early approaches have the problem of selecting an appropriate trained range. Very recently, a so-called "an improved neural network approach" proposed by Lin and Chen (2006) is based on the combination of an ANN to find the coordinate of match point from drawdown ratios and the Theis solution. It avoids the problem of selecting an appropriate trained range, determines the aquifer parameter values more accurately, and has a simpler structure and needs less training time, compared to previous works.

In this paper, first, a modified version of the Lin–Chen ANN approach is presented by applying LM training algorithm. This significantly improved the performance of the network and reduced the convergence criterion from $10^{-3}$ to $10^{-9}$, which was non-attainable by gradient descent algorithm in the original network. Moreover, the training sets have been normalized to have zero mean and unit variance. Afterward, the PCA technique was applied to reduce the dimension of input data from $N \times 1$ to only one by considering a minimum fraction variance of 0.001.

The method of PCA, performed with MATLAB R14 (Math Works, Inc. 2004) allowed identification of associations between training patterns. The eigenvalues and eigenvectors extracted from the correlation matrix provide the basic solution to the multivariate problem. The eigenvalues represent the fraction of variance contributed by each eigenvector or PCs. It showed that the first principal component (PC1) explains 99.911% of the variance in the original training data set, and the second principal component (PC2), explains 0.089% of variance. Therefore by applying the PCA technique, we could reduce the dimension of original training data set from $N - 1 \times M$ matrix (where $N$ in the number of drawdowns and $M$ is the number of training patterns, i.e., 10,205), to $1 \times M$ vector by considering 0.001 as minimum fraction variance. This vector is equivalent to the first PC and is applied in the training stage. Further, a transformation matrix of $1 \times (N - 1)$ elements is formed which consists of the eigenvectors of the input covariance matrix. All subsequent inputs to the previously trained network at testing/verification stage need to be transformed using this transformation matrix. Therefore, a pumping drawdown test data consisted of $N$ records is converted to a columnar $(N - 1) \times 1$ vector of drawdown ratios based on the Lin–Chen (2006) training approach. Then this vector is normalized by considering the mean and variance of training data set. The transformation matrix $(1 \times N - 1)$ is multiplied by the normalized data and the result is a single input value (equivalent to the match point in the transformed space) to the trained network. Because of simplifying input space dimension, we could reduce the number of hidden layer neurons from 32, in the trained original and modified Lin–Chen ANNs, to a minimum of eight while preserving the desired accuracy. Finally, the structure of the proposed network is fixed with the topology of $1 \times 8 \times 1$, which is not changed through the network application for any individual case study.

Therefore, the advantages of the proposed ANN over the original Lin–Chen are as follows:

- It uses the LM training algorithm that improves the performance of the network from $10^{-3}$ to $10^{-9}$ (in terms of convergence criterion) and reduces the training time.
- Due to the application of PCA, it has a simpler topology of $1 \times 8 \times 1$, instead of $(N - 1) \times 32 \times 1$ in the original and modified Lin–Chen networks. This reduces the training time further (up to 10 times).
• It is independent of the number of drawdown records \((N)\), which may increase considerably in continuous data logger systems with high frequency intervals prevalent in water resources.

• It was verified with extremely noisy drawdown data (pumping variations on the early time and systematic noise in the late time records in the second case study, Eq. 3.6 in p. 64 of Kruseman and de Ridder, 1992). This proves the better generalization and reliability of the proposed network.

• The procedure presented in this research may be used to model other well functions.

• The proposed simple ANN appears to be an alternative to previous ANNs and traditional type curve matching techniques.

Acknowledgements

The authors gratefully acknowledge the financial support provided by the Ministry of Energy, Water Research Institute (WRI), and Shiraz University, Iran. Useful comments and suggestions given by the editor and anonymous reviewers are greatly appreciated.

References


Theis, C.V., 1935. The relationship between the lowering of the piezometric surface and the rate and duration of discharge of a well using ground-water storage. Transactions American Geophysical Union 16, 519–524.


