International Journal of Applied Mathematics and Engineering Sciences
Vol. 4 No. 1 (June, 2019)

ESTIMATION OF CONTAMINANT TRANSPORT IN A LAYERED AQUIFER USING ARTIFICIAL NEURAL NETWORKS AND FINITE DIFFERENCE METHOD

S. S. Kadam, S. K. Das & A. S. Warke

Received: 14th December 2018 Revised: 18th March 2019 Accepted: 22nd April 2019

***ABSTRACT*:** In this paper, we estimate the contaminant transport in a two-layered aquifer by employing Artificial Neural Network (ANN) that uses Multi Layer Perceprton (MLP) architecture. The input data for ANN was generated from the output of Finite Difference (FD) method based on advection-dispersion equation in coupled form. Employing Alternate Direction Implicit (ADI) scheme, coupled transport equations were solved numerically to generate the inputs. Using back propagation technique, ANN model was trained corresponding to inputs of upper and lower layers of contaminant concentration for fixed values of parameters and time period. This trained ANN model was then used to estimate the contaminant transport beyond the specified training period. To measure the error between ANN predicted contaminant values and those computed by the Finite Difference model, we have used three different error measures, namely, the root mean square error, the mean absolute error, and the percent mean relative error for proper interpretation of the results. The results based on ANN and the FD models have shown good agreement. Based on this methodology, it is possible to show that a three-layer feed-forward back propagation ANN model with a nonlinear differentiable log-sigmoid transfer function in both the hidden and output layers and a variable learning rate, can replace traditional transport modeling.

KEYWORDS: Contaminant transport; Artificial Neural Network; Aquifer; ADI scheme; Layered model; Advection-diffusion; Root Mean Square Error; MLP architecture.

NOMENCLATURE

c_u contaminant concentration in upper layer

c_l contaminant concentration in lower layer

D_{ux}, D_{uy} dispersion coefficient along horizontal and vertical directions in upper layer

D_{lx}, D_{ly} dispersion coefficient along horizontal and vertical directions in lower layer

u_u, u_l seepage velocities in upper and lower layer respectively (horizontal direction)

v_u, v_l seepage velocities in upper and lower layer respectively (vertical direction)

k exchange coefficient

β reaction parameter

x_j input layer

a_i hidden layer

y_j output layer

$\delta\theta_i, \delta\tau_j$ updated threshold values

v_{ih}, w_{ji} weights

d_j, e_j errors in the output and hidden layers

E_i total error

1. INTRODUCTION

Traditionally, the extent of contaminant spread is predicted by solving advection-dispersion equation analytically for simplified problem, otherwise by using numerical model [2, 26]. As the number of unknown parameters become large, the model error becomes significant, particularly when the model equations are unsteady, non-linear or coupled in form. For such complex problem the performance of the numerical method depends upon the scheme applied, and its inherent error control mechanism. The groundwater models are usually developed for various process-based components to take into account all relevant details, i.e., advection, dispersion, sorption, biodegradation etc, and their complex interactions. However, variability of hydro-geologic parameters further complicates and makes it most uncertain to quantify. Field measurement, which is considered to be an integral part to verify the model results, is either not adequate or poorly represented [9]. In recent years, lot of concerns on numerical solution has focused on predicting accurate solution that is somewhat free from unphysical numerical oscillations. The increased applications of higher order numerical schemes applicable to nonlinear systems of advection-diffusion-reaction type problems tend to show unphysical oscillation in the presence of steep concentration gradient [10]. Another drawback common to these traditional techniques is that they suffer from cancellation effect when compensating errors in two or more coefficients produce reasonable prediction. Carrera [3] described the state of the art of the inverse problem applied to flow and solute transport equations and emphasized the difficulty in modeling due to: (i) increased complexity of the problem, (ii) conceptual limitations of the transport processes in groundwater environment, (iii) drawback common in the traditional technique due to cancellation effects. According to this motivation, we apply ANN technique to estimate and compare contaminant transport in a coupled unsteady two-layered aquifer.

In recent years, Artificial Neural Network (ANN) is being increasingly applied as an efficient mathematical tool to represent complex relationships in many branches of hydrology [11, 14-17, 20, 22]. ANN's flexible structure can provide good estimation to various problems in hydrology such as water quality modeling, stream flow forecasting, groundwater modeling and precipitation forecasting, etc, [3]-[5]. Yeh [23] has reported various techniques to solve inverse problem of parameter evaluation in groundwater. In particular, the back propagation algorithm as a theoretical framework have resulted wide application in various civil engineering problems [1].

In the present study, we consider a numerical model that serves as a template to describe the contaminant transport in a two-layered system wherein the data generated by solute transport model is being trained by the ANN model and applied to estimate future scenarios when interface and other reactive boundary conditions are considered. The main objectives of the present study are to obtain realistic ANN simulation and to estimate the extent of contaminant spread, when rain infiltration and chemical reaction take place simultaneously at the upper and lower layer boundaries respectively. The present study may find application in estimating leachate movement due to the leakage of underground storage tanks.

2. MATHEMATICAL FORMULATION

We consider two-dimensional, two-layer advection-diffusion equation in a finite computational domain described by Cartesian coordinate system where each layer is homogeneous and isotropic and connected through patchy, pervious interface (Figure1).

The governing equations describing the sub-surface transport of a non-conservative solute through a saturated and non-deformable aquifer with reactive lower layer boundary can be represented as [21].

$$D_{UL}\frac{\partial^2 C_U}{\partial X^2}+D_{UT}\frac{\partial^2 C_U}{\partial Y^2}-U_U\frac{\partial C_U}{\partial X}-V_U\frac{\partial C_U}{\partial Y}-R_U\frac{\partial C_U}{\partial T}=K\left(C_U-C_L\right) \quad (1)$$

$$D_{LL}\frac{\partial^2 C_L}{\partial X^2}+D_{LT}\frac{\partial^2 C_L}{\partial Y^2}-U_L\frac{\partial C_L}{\partial X}-V_L\frac{\partial C_L}{\partial Y}-R_L\frac{\partial C_L}{\partial T}=K\left(C_L-C_U\right) \quad (2)$$

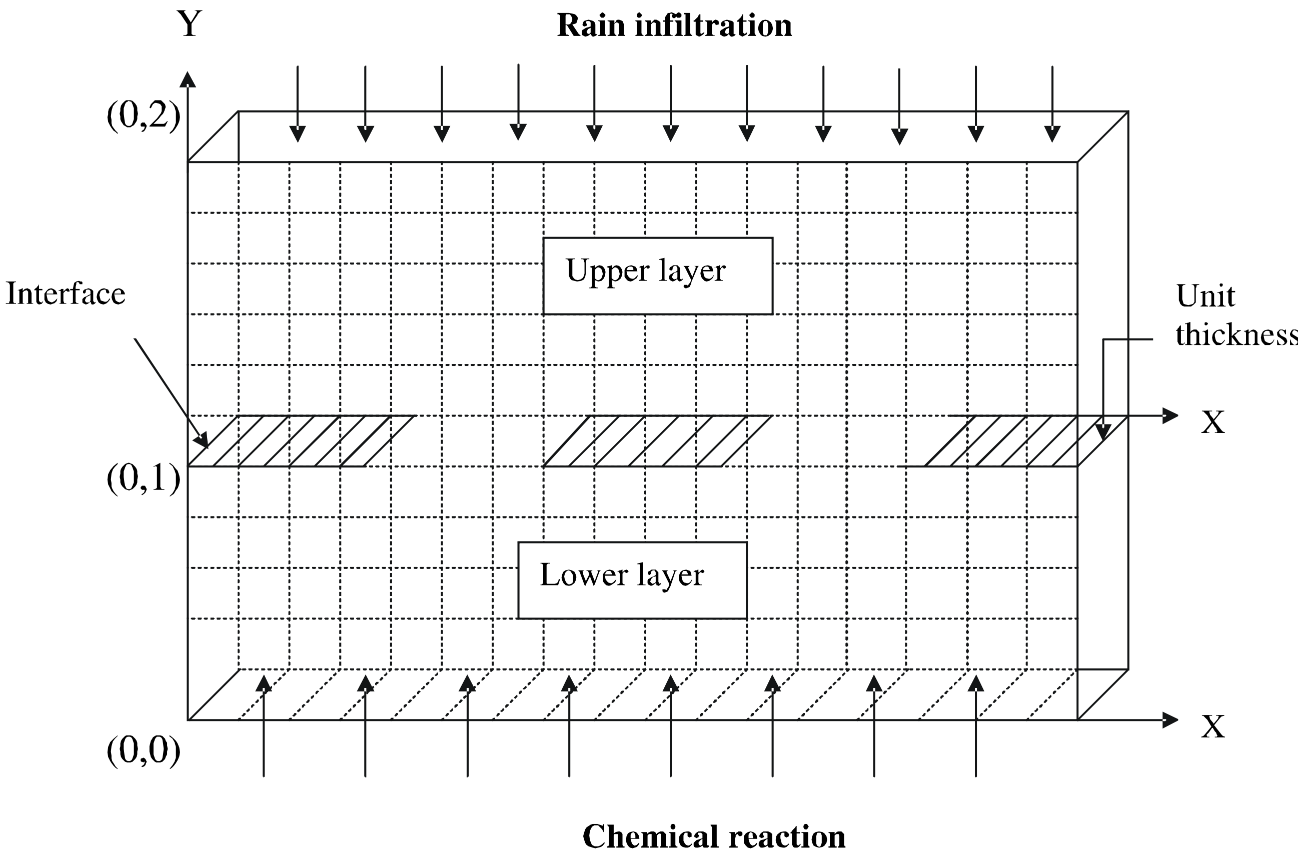


Figure 1: Schematic Diagram of Layered Aquifer

where

$$X = \frac{x}{L}, Y = \frac{y}{L}, C_U = \frac{c_u}{C_0}, C_L = \frac{c_l}{C_0}, T = \frac{tV_L}{L}, V_U = \frac{v_u}{V_L}, V_L = \frac{v_l}{V_L}$$

$$D_{UL} = \frac{D_{ux}}{LV_L}, D_{UT} = \frac{D_{uy}}{LV_L}, D_{LL} = \frac{D_{lx}}{LV_L}, D_{LT} = \frac{D_{ly}}{LV_L} \text{ and } K = kL \tag{3}$$

Here, X-axis is along the longitudinal direction (aquifer length) and Y-axis is vertically upward (towards upper surface). The governing equations (1)-(2) are written in dimensionless form where C_U and C_L are the solute concentrations, U_U and U_L are seepage velocities along X-direction, V_U and V_L are seepage velocities along Y-direction, D_{UL} and D_{LT} are the dispersion coefficients (L^2/T) along X-direction, D_{UT} and D_{LT} are the dispersion coefficients (L^2/T) along Y-direction, R_U and R_L are retardation factors for upper and lower layers respectively. K is the exchange coefficient between upper and lower layer and C_0 is the reference concentration.

Solution of transport equation requires a specified initial concentration and appropriate boundary conditions at the top and base of the soil column. Hence, corresponding to the model equations (1) and (2), the dimensionless form of initial and boundary conditions can be specified as

$$C_U(X,Y,0) = C_{Uinput} \text{ and } C_L(X,Y,0) = C_{Linput} \tag{4}$$

$$\lim_{X\to 0}\frac{\partial C_U}{\partial X}=0 \text{ and } \lim_{X\to 0}\frac{\partial C_L}{\partial X}=0 \tag{5}$$

$$\lim_{X\to \infty}\frac{\partial C_U}{\partial X}=0 \text{ and } \lim_{X\to \infty}\frac{\partial C_L}{\partial X}=0 \tag{6}$$

$$\lim_{Y\to 1}\frac{\partial C_U}{\partial Y}=0 \tag{7}$$

The interface boundary condition between the upper and lower layers can be posed as

$$C_U\left(X,0,T\right)=C_L\left(X,1,T\right) \text{ for } X_3 \le X \le X_4 \text{ and } X_5 \le X \le X_6 \tag{8}$$

The boundary condition at the lower layer bed can be described as [13];

$$\lim_{Y\to 0}\left(\frac{\partial C_L}{\partial Y}+\beta C_L\right)=0 \tag{9}$$

Equation (9) represents mixed boundary condition to specify first order chemical reaction. This condition not only provides zero gradient condition but also indicate depletion at the lower layer bed. The governing equations (1) and (2) are coupled due to the presence of source or sink terms and are not amenable to closed form solution owing to interface and reactive boundary conditions prescribed by the equations (8) and (9).

3. NUMERICAL COMPUTATIONS TO GENERATE ANN INPUT

In order to solve equations (1) and (2) together with the prescribed boundary conditions (4)-(9), we apply Alternate Direction Implicit (ADI) - finite difference method based on Doglas-Rachford scheme [6]. The finite difference analogues of the governing equation using double sweep technique are represented by

$$P\,C_{i,j-1}^{n+1}+Q\,C_{i,j}^{n+1}+R\,C_{i,j+1}^{n+1}=S \qquad \textit{Y - direction} \tag{10}$$

and

$$U\,C_{i-1,j}^{n+2}+V\,C_{i,j}^{n+2}+W\,C_{i+1,j}^{n+2}=Z, \qquad \textit{X - direction} \tag{11}$$

where *(P, Q, R, S)* and *(U, V, W, Z)* are the known coefficients at $\Delta t/2$ and Δt time steps respectively. The finite difference representation of the initial and boundary conditions are

$$C_{i,j}^{U}\Big|_{t=0}=C_{input}^{U} \text{ and } C_{i,j}^{L}\Big|_{t=0}=C_{input}^{L} \quad \textit{initial condition}$$

$$\left[\frac{C_{i,j}^{n+1}-C_{i,j-1}^{n+1}}{\Delta y}\right]_{j=NY}^{U}=0\,, \qquad \textit{at the upper surface}$$

$$\left[\frac{C_{i,j}^{n+2}-C_{i-1,j}^{n+2}}{\Delta x}\right]_{i=0}^{U,L}=0 \;\textit{ and }\; \left[\frac{C_{i,j}^{n+2}-C_{i-1,j}^{n+2}}{\Delta x}\right]_{i=NX}^{U,L}=0, \qquad \textit{at the left and right boundary}$$

$$\left[(1+\beta\,\Delta y)C_{i,j}^{n+1}-C_{i,j-1}^{n+1}\right]_{j=0}^{L}=0 \qquad \textit{at the bottom bounday of lower layer}$$

and at the interface boundary

$$C_{i,0}^{U}=C_{i,NY}^{L}\;,\; NX3\le i\le NX4 \text{ and } NX5\le i\le NX6 \tag{12}$$

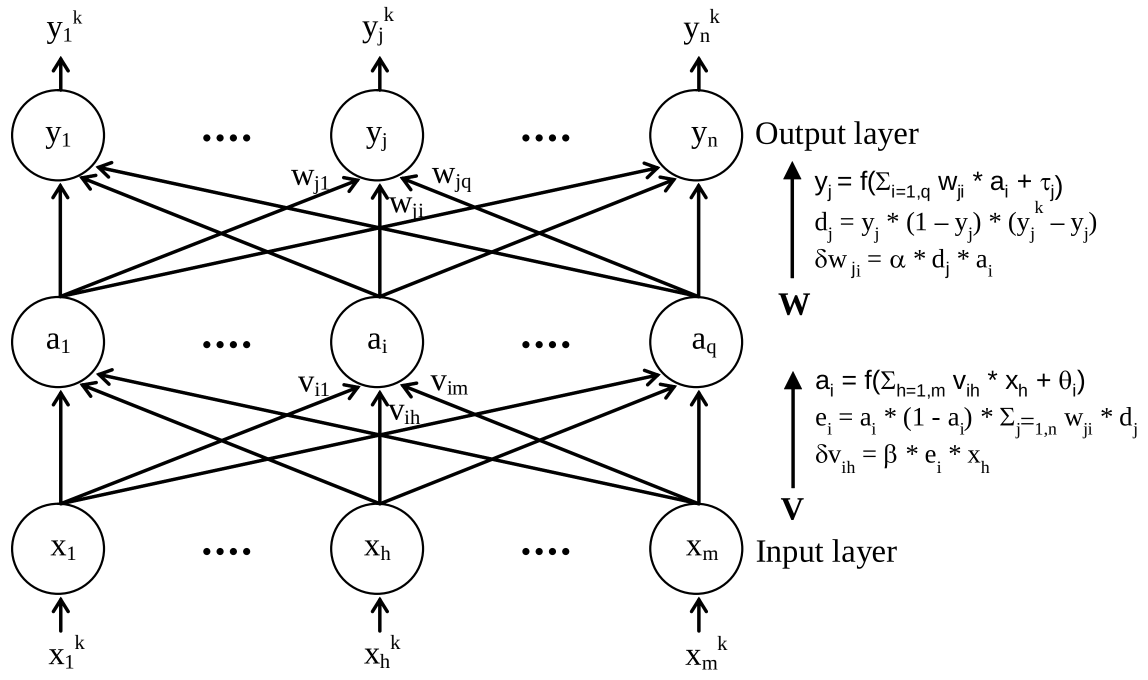
where the superscripts in (12) indicate the layer associated corresponding to the boundary grid points. It may be noticed that the resulting finite difference equations (10) and (11) are simultaneous linear algebraic equations with tri-diagonal coefficient matrix. We apply double sweep algorithm consisting of two steps involving the solution of tri-diagonal sets of linear equations along the lines parallel to *X* and *Y*-axis at the first and second time steps respectively. In the first time step, X sweep is carried out by taking the implicit difference formulation for $\frac{\partial^2 C_i}{\partial Y^2}$ along *Y*-direction and explicit difference formulation for $\frac{\partial^2 C_i}{\partial X^2}$, *i=L, U*. In the second time step, the above difference formulation is reversed. The difference scheme described here is unconditional stable and provides second order accuracy in time and spatial directions [6]. In order to control numerical oscillation and dispersion grid Peclet and Courant numbers are checked and found to be well within the limits. To obtain the ANN inputs, we perform numerical simulation by assuming the values of various parameters as [21]:

$D_{UL} = 0.5$, $D_{UT} = 0.05$, $D_{LL} = 0.5$, $D_{LT} = 0.05$, $R_U = 1.0$, $R_L = 1.0$, $\Delta T = 0.0025$, $K = 1.0$ and V_U is the rain infiltration velocity in the upper layer whereas V_L, U_U and U_L are zero in the respective layers. Initially, we consider that lower layer is free from contaminant concentration (C^L_{input} =0.0) whereas in the upper layer it is assumed to be unity (C^U_{input} = 1.0). The model area covers a distance 500m along longitudinal direction and 200m along vertical direction whose non-dimensional distance varies between 0 to 5 along the length and 0 to 1 along the depth of aquifer for each layer. A square grid cell with $Dx = Dy = 0.1$, equivalent to dimensional distance of 10m is considered for which the total number of grids along *(X, Y)* for each layer become (50,10). The interface boundary conditions are described between the grids (10-20) and (30-40) along *X* direction, where exchange of flow and transport processes take place through leaky pervious medium. The main objective of this procedure is to obtain concentration values in each grid points in the upper and lower layers indicating the contaminant spread due to the effect of rain infiltration in the upper layer and first order reactive boundary condition imposed at the bottom boundary of lower layer [13]. Prior to make numerical computation for rain infiltration and boundary reaction, we verify and calibrate the model with the available result of Warke and Das [21]. The computations were carried out from one year (T=0.365) to ten years (T=3.65) period to access the long-term effect and year wise outputs are stored for all the grid points. These simulated data constitute necessary input for the ANN model.

4. ARTIFICIAL NEURAL NETWORKS

Artificial neural networks are computational models characterized by a series of processing elements (PEs), called neurons or nodes, which enable them to process patterns within the data. These neurons interact with each other through weighted connections or weights, and work in unison to solve a computational problem by performing two important processes: (i) internal activation; and (ii) transfer function. In the internal activation process, the values of incoming information are multiplied by the corresponding connection weights to produce an internal activation value using a summation function. On the other hand, the transfer function calculates the activation level of neurons from their corresponding internal activation values. The neurons in a neural network are usually arranged in layers, and a typical neural network consists of one or more such layers.

A layer is associated with three functions, namely, summation, transfer (activation), and output, which are used to compute the outputs of all neurons in that layer. We present the patterns to the neural network through the input layer where neurons communicate with the neurons in one or more hidden layers and actual processing is done via a system of weighted connections. The hidden layers are then linked to an output layer where the output of the neural network is produced. Amongst many available neural network architectures, we have used the multi-layer perception architecture (MLP) as depicted in Figure 2. Although there are many different types of learning rules, the most commonly used delta rule under the class of back propagation techniques has been



y_1^k
y_j^k
y_n^k
y_1
y_j
y_n
Output layer
w_{j1}
w_{jq}
w_{ji}
$y_j = f(\Sigma_{i=1,q}\, w_{ji} * a_i + \tau_j)$
$d_j = y_j * (1 - y_j) * (y_j^k - y_j)$
$\delta w_{ji} = \alpha * d_j * a_i$
W
a_1
a_i
a_q
v_{i1}
v_{im}
v_{ih}
$a_i = f(\Sigma_{h=1,m}\, v_{ih} * x_h + \theta_i)$
$e_i = a_i * (1 - a_i) * \Sigma_{j=1,n}\, w_{ji} * d_j$
$\delta v_{ih} = \beta * e_i * x_h$
V
x_1
x_h
x_m
Input layer
x_1^k
x_h^k
x_m^k

Figure 2: A Three Layer Fully Interconnected MLP Architecture

applied in our study. With the delta rule, learning becomes a supervised process that occurs with each cycle or 'epoch' through a forward activation flow of outputs, and backward propagation of errors for weight adjustments. For a given input pattern, the MLP neural network computes the output and compares it with the expected output pattern, and then adjusts the network connection weights in order to minimize the error between the expected and the computed output. Within each hidden layer, a typical sigmoid activation function is adopted, which polarizes the network activity and helps it to stabilize.

The formal algorithm to train the back propagation neural network is illustrated below and is based on the work of Hechst-Nielsen [8] and Simpson [19]. An implementation of this algorithm can be found in Demluth [7] and Masters [12].

1. Randomize the network weights in the range [-1, 1]

2. For each pattern $(X_k, Y_k) = \langle (x_1^k, x_2^k, \ldots, x_m^k), (y_1^k, y_2^k, \ldots, y_n^k) \rangle, \quad k = 1,2,\ldots,p$, do

   (i) Present input pattern to PEs in the input layer

   (ii) Compute new output values of PEs in the hidden and output layers using

$$a_i = f\left(\sum_{h=1}^{m} v_{ih} x_h + \theta_i\right) \qquad for\ i = 1,2,\ldots,q \tag{13}$$

$$y_j = f\left(\sum_{i=1}^{q} w_{ji} a_i + \tau_j\right) \qquad for\ j = 1,2,\ldots,n \tag{14}$$

where Q_i and T_j are threshold values, and $f(x) = 1/(1+e^{-x})$ is the sigmoid transfer function

   (iii) Estimate the "error term" between computed and desired output values of PEs in the output and hidden layers using

$$d_j = y_j\left(1-y_j\right)\left(y_j^k - y_j\right) \qquad for\ j = 1,2,...,n \tag{15}$$

$$e_i = a_i\left(1-a_i\right)\sum_{j=1}^{n} w_{ji} d_j \qquad for\ i = 1,2,...,q \tag{16}$$

3. Update the threshold values

$$\delta\tau_j = \alpha\ d_j \tag{17}$$

$\delta\theta_i = \beta\ e_i$, where α and β are learning rates (18)

4. Repeat steps 2-3 until error d_j is sufficiently low for each *j = 1,2,…,n* and *k = 1,2,..,p*

The above-mentioned algorithm performs a gradient descent procedure to obtain global minimum along the steepest vector of the error surface. As the error surface could be hyper-paraboloid in nature, but rarely smooth, the solution space contains irregular solution vectors, which may cause the network to settle down in a local minimum. In such case the errors made by network are given by

$$E = E_j + E_i \tag{19}$$

where E_j is the total error in neurons in the output layer, and E_i is the total error in neurons in the hidden layer. These error components can be expressed as

$$E_j = \sum_j \left(y_j^k - y_j\right) \tag{20}$$

$$E_i = \sum_{j=1}^{n} w_{ji} d_j \tag{21}$$

where $d_j = y_j\left(1-y_j\right)\left(y_j^k - y_j\right)$ is the error term in neurons of the output layer. This learning procedure attempts to modify the network weights towards global minima using built-in mathematical expressions to control the speed (beta-coefficient) and momentum.

A large number of individual runs were taken to determine the best solution, since the nature of the error space could not be determined a priori.

5. TRAINING AND TESTING

In the present application, we have employed a supervised back propagation neural network comprising three layers, simulated with the MATLAB@ Neural Network Toolbox (Release-12) [7] loaded on a personal computer running under Windows@ 2000 operating system. In the training stage, a set of training data with input examples and their corresponding desired outputs was prepared. The network weights, which eventually store the learned patterns, were initially set to random values. During the learning process, the example inputs are given to the network and the output of each neuron in the network is calculated. After comparing the computed and desired outputs, an error term is calculated, which is used to change the network weights in an iterative manner [18]. The processing of information is done by the log-sigmoid transfer function in both the hidden and the output layers. The log-sigmoid function generates outputs between 0 and 1 as the neuron's input goes from negative to positive infinity. The advantage of using a log-sigmoid function over a threshold-type function is that the former is continuous and differentiable, which allows the gradient of the error to be used in updating the weights. We have used a variable learning rate during the training process. The use of variable learning rate improves the convergence of the network by ten to one hundred times than the standard steepest descent methods with fixed learning rate. The training data comprised contaminant values in the upper and lower layer of the aquifer taken over several years.

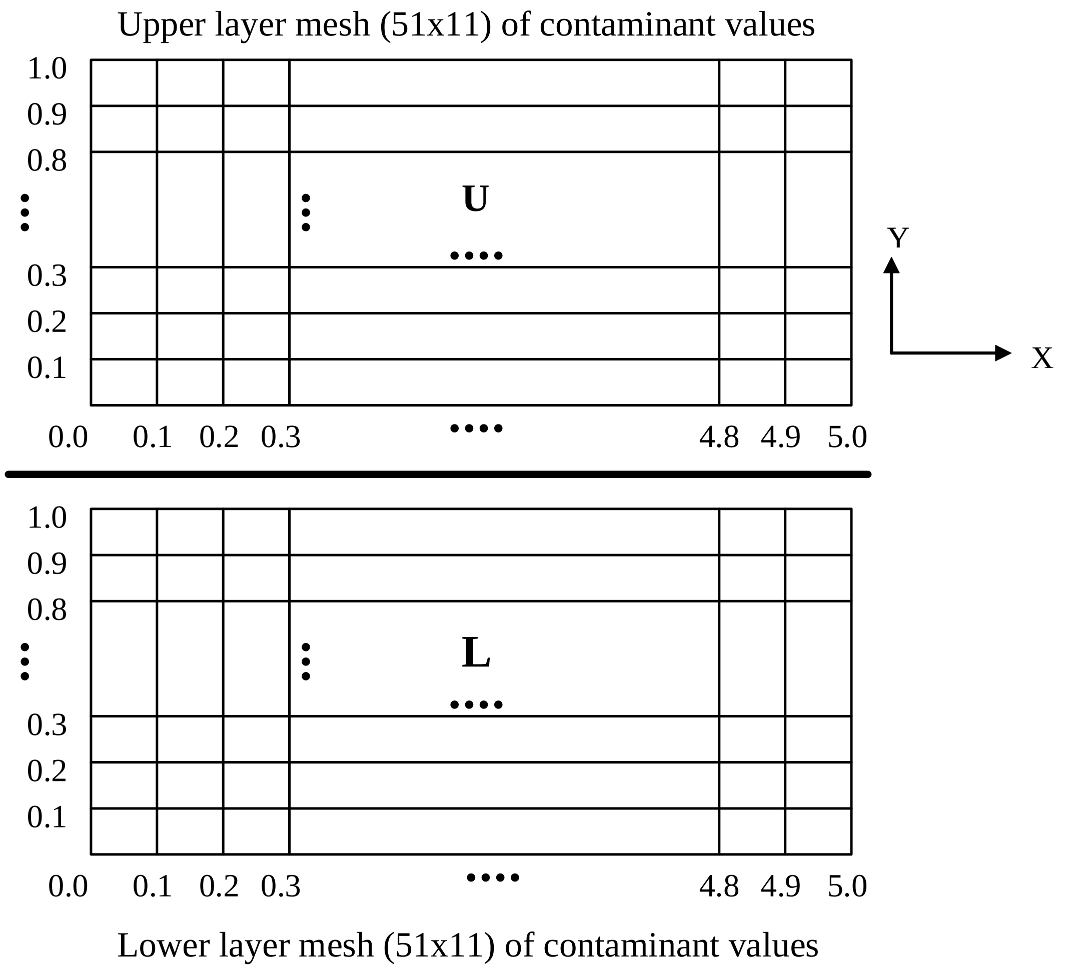

Upper layer mesh (51x11) of contaminant values
1.0
0.9
0.8
0.3
0.2
0.1
0.0
0.1
0.2
0.3
4.8
4.9
5.0
U
Y
X
L
Lower layer mesh (51x11) of contaminant values

Figure 3 (a): Schematic Diagram of Layer Wise Computational Grid Points

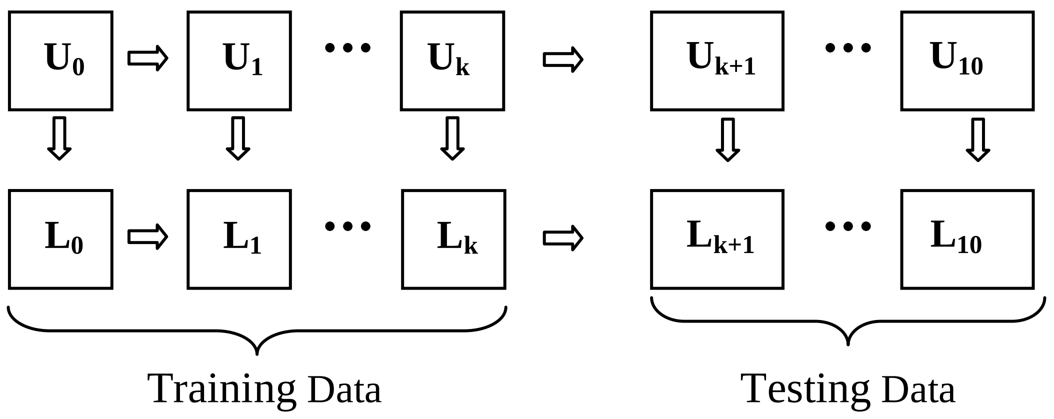


Figure 3 (b): Training and Testing Processes

For a particular year, two sets of 561 (51x11) contaminant values were taken, one in the upper layer and the other in lower layer of the aquifer as shown in Figure 3(a). The layers were separated by an impervious boundary, but also had patchy finite strips of pervious interfaces. The entire data comprised 11 pairs of upper and lower layer contaminant values representing different years as shown in Figure 3(b). The contaminant values were normalized to unity at their maximum values. The neural network was separately trained with 6, 7, and 8-years of contaminant data (k = 6, 7, 8). Then each trained ANN was subsequently tested with 4, 3, and 2-year contaminant data (comprising grid points in the upper and lower layer respectively). Figures 4-6 show the contaminant distribution in the presence of rain infiltration (V_u = 0.1, 1.0) and boundary reaction (beta = 5,10) after two, six and ten years of simulation. The symmetric nature of contaminant plume under the above-mentioned condition provides a good check for numerical simulation.

In order to study the growth of error in prediction with increasing prediction times (years), the back propagation neural network model was separately trained with different sets of training data (comprising pairs of the upper and lower layer contaminant values) to obtain three separate trained networks. Each trained network was tested with pairs of contaminant values for the remaining years as shown in Figure 3(b). For example, the neural network model was initially trained with pairs of the upper and lower layer contaminant values for first six years to obtain the trained network ANN1, while the pairs of contaminant values for the remaining four years were used to test this trained network. Similarly, the neural network was retrained with 7-year and 8-year pairs of the upper and lower layer contaminant values to obtain trained networks ANN2 and ANN3, respectively. The trained ANN2 and ANN3 networks were tested with the subsequent 3-year and 2-year pairs of the upper and lower layer contaminant data, respectively.

For measuring the error between the ANN predicted contaminant values and those computed by the Finite Difference model, we have used three different error measures, namely, the root mean square error (RMSE), the mean absolute error (MAE), and the percent mean relative error (PMRE), the expressions for which are as given below,

$$RMSE = \sqrt{\frac{\sum_{j=1}^{n}\left(y_j^k - y_j\right)^2}{N}}; \quad MAE = \frac{\sum_{j=1}^{n}\left|\left(y_j^k - y_j\right)\right|}{N}; \quad PMRE = 100\left(\frac{\sum_{j=1}^{n}\left|\left(y_j^k - y_j\right)\right| \Big/ y_j}{M}\right) \tag{22}$$

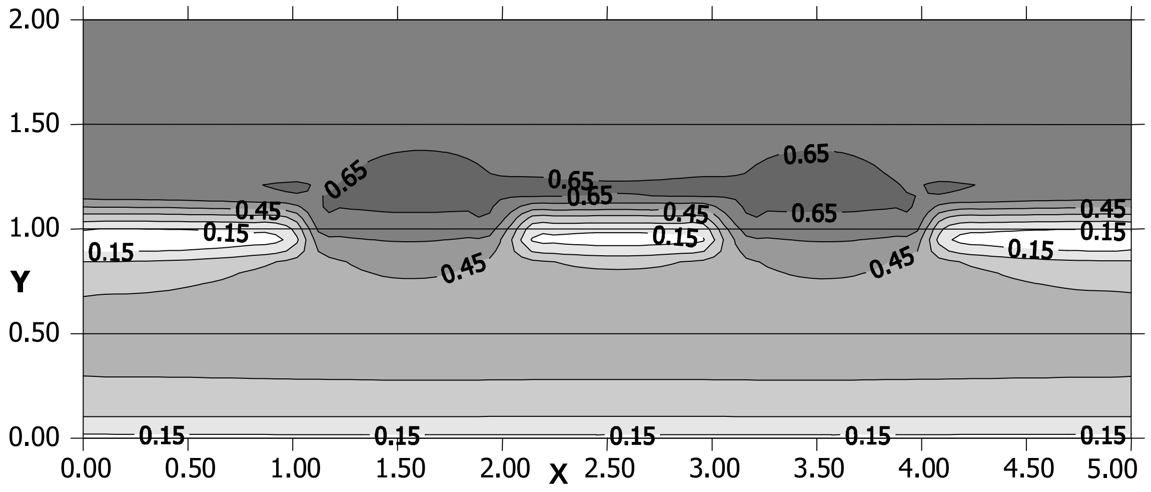


Figure 4 : Contaminant Dispersion in Layered Aquifer with Reaction Beta = 10 at Vu = 1 after 2 Years

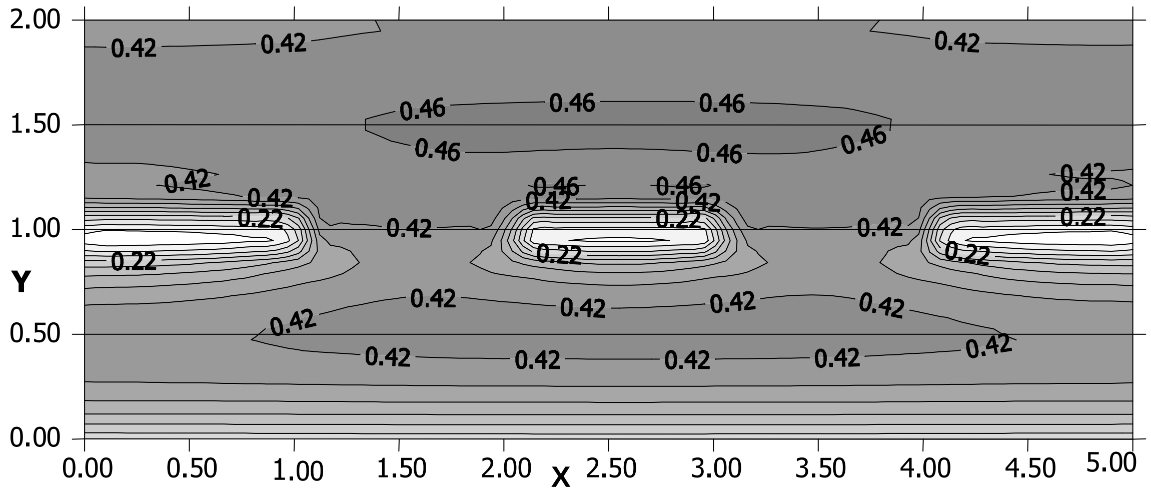


Figure 5 : Contaminant Dispersion in Layered Aquifer with Reaction Beta = 5 at Vu = 0.1 after 6 Years

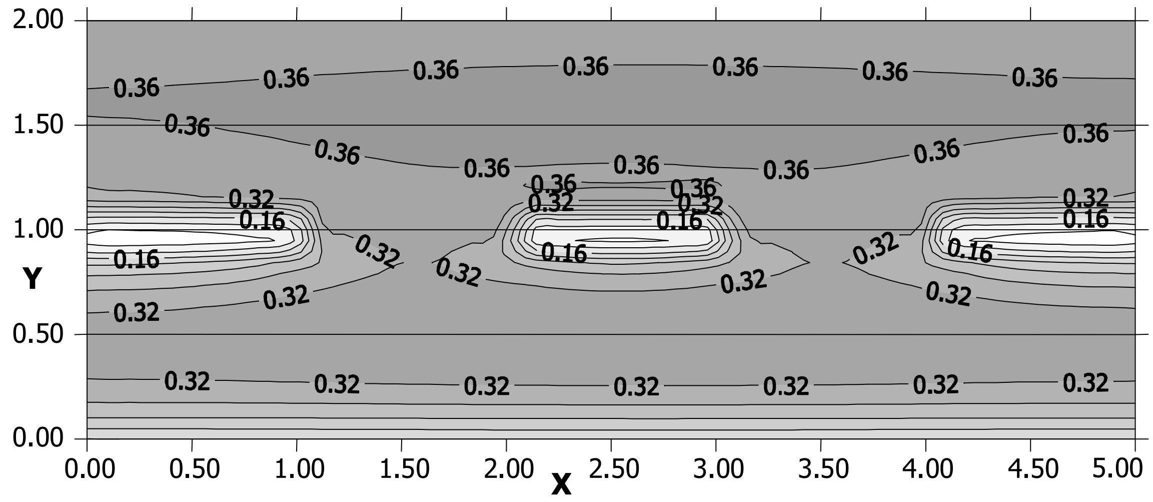


Figure 6 : Contaminant Dispersion in Layered Aquifer with Reaction Beta = 5 at Vu = 0.1 after 10 Years

where, y_j^k represents the ANN computed contaminant value, y_j represents the corresponding value computed by FD method, *N* is the total number of contaminant values in a layer for particular year, and *M* is the total number of non-zero $(y_j \neq 0)$ contaminant values in a layer for particular year computed by the FD method.

The RMSE and MAE are measured in the same units as data and are therefore easy to understand. Both represent the size of a typical error between the contaminant values predicted by the neural network and the corresponding values computed by the FD model. However, most researchers prefer unit-free measures for comparing methods [24]. The PMRE, unlike RMSE and MAE, is a unit-free measure that gives percent mean relative error between the computed and expected values. Moreover, RMSE has low reliability [25], nevertheless, it has been widely used for comparing forecasting methods [24]. The three error measures were taken in order to analyze the comparative results form different perspectives, take care of sensitivities due to small changes in large number of data series, and draw reliable and proper conclusions.

The learning curve in Figure 7 shows how the total sum-squared error (SSE) converges in the process of iterative learning for ANN1. The learning curves for ANN2 and ANN3 are similar. By total sum-squared error we mean $\sum_p \sum_i E_i^2$, where *p* ranges over all the training patterns, and *i* ranges over the entire network neurons (E_i is error at the i^{th} neuron given by Equations 20-21). The learning process terminates either after a certain number of runs through all the training data (each run through all the training data is called an epoch), or when the total sum-squared error reaches some predefined target value or goal. In this study, we fixed the target SSE (Goal) to 0.01, and the maximum number of epochs to 5000. The number of epochs required to achieve the target SSE of 0.01 while training ANN1, ANN2, and ANN3 were 3063, 3994, and 4998, respectively.

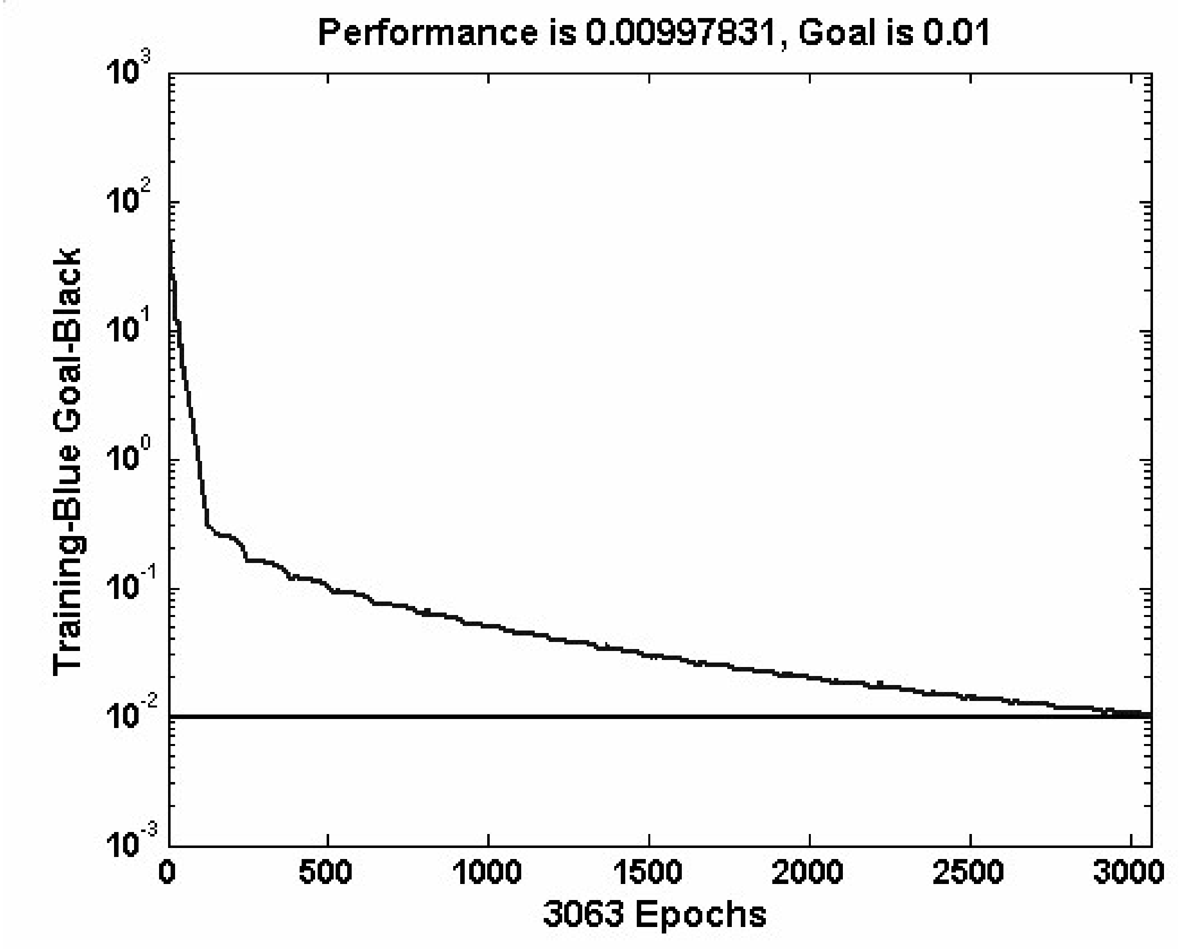


Figure 7 : Convergence of Training curve with respect to the total SSE observed while training ANNI neural network

In the testing stage, test patterns are given to the corresponding trained networks and information in the form of stored weights is used to determine the class of a test pattern in terms of training classes. The performance of the ANN method was judged by comparing the contaminant values determined from the ANN and by the FD model. Table 1.0 displays values of different error measures that were applied to the contaminant values predicted by the three ANNs and the corresponding values computed by the FD method. Since ANN1 was trained with pairs of the upper and lower layer contaminant values for initial six years, it was used to predict the contaminant values for the remaining four years (i.e., for 7th, 8th, 9th, and 10th year). The entries in Column I ('ANN1 prediction errors') of Table 1.0 display prediction errors in the contaminant values computed by ANN1 relative to those computed by the FD method. Similarly, as ANN2 and ANN3 neural networks were respectively trained with initial 7-year and 8-year contaminant data, they were used to predict the contaminant values for the remaining four and three years, respectively (i.e., for 8th, 9th & 10th year; and 9th & 10th year). The entries in column II and column III of Table 1.0 display prediction errors for ANN2 and ANN3, respectively.

Table 1
Prediction Errors in Contaminant Values Computed by ANNs and the FD Method

Prediction for year	*ANNI prediction errors (I)*			*ANN2 prediction errores (II)*			*ANN3 prediction errors (III)*		
	RMSE	*MAE*	*PMRE*	*RMSE*	*MAE*	*PMRE*	*RMSE*	*MAE*	*PMRE*
7	0.0079	0.0073	1.71 %						
8	0.0157	0.0144	3.47 %	0.0078	0.0071	1.67 %			
9	0.0248	0.0228	5.60 %	0.0149	0.0136	3.30 %	0.0072	0.0065	1.56 %
10	0.0347	0.0320	8.04 %	0.0233	0.0213	5.29 %	0.0135	0.0122	3.01 %

The column entries in Table 1.0 show how the error in prediction gradually increases with increase in prediction times (years). However, as the number of training samples is increased, the error in prediction decreases significantly with increase in prediction times. For example, the contaminant values predicted for the 10th year show RMSE, MAE, and PMRE as 0.0347, 0.0320, and 8.04% using ANN1 (trained with 6-year contaminant data) relative to corresponding values computed by the FD method. The same sample when predicted using ANN2 (trained with 7-years data) and ANN3 (trained with 8-year data), shows decrease in prediction errors as seen from the entries in last row of Table 1.0 under column II and III, respectively. Similar trend can be observed while predicting the contaminant data for 9th year. Both the 9th and 10th year contaminant data was used for testing all the three trained ANNs, since it did not form part of the training data of the three ANNs. The decrease in prediction errors with increase in training samples used for training the networks can be observed by comparing entries under column I of Table 1.0 with the corresponding entries under column II and column III.

The 3-D plots of the contaminant values computed by the ANN method and the FD model together with the 3-D plots of absolute error between the ANN predicted and model computed contaminant values are depicted in Figures 8-10. Figure 8 shows the 3-D scatter plot of the contaminant values computed by the trained ANN1 and the FD model for four subsequent years (i.e., for 7th, 8th, 9th and 10th year). As seen from the Figure 8, the contaminant values predicted by ANN1 are almost same as those computed by the FD model. The plot of the absolute error between contaminant values computed by ANN1 and the FD model for four subsequent years is shown in Figure 9. From Figure 9, it can be observed that the absolute error values are significantly small; the corresponding MAE values are enlisted in column I of Table 1.0. Figure 10 displays the plot of absolute error between the contaminant values computed by ANN2 and the FD model (first two plots), and ANN3 and the FD model (last two plots) for the 9th and 10th year. The two 3-D plots depicted in the last row of Figure 9 together with the 3-D plots depicted in Figure 10 show the manner in which the prediction errors decrease with increase in training data. The contaminant values for the 9th and 10th year can be predicted using all the three trained ANNs (i.e., ANN1, ANN2, and ANN3).

The 3-D plots show that the growth of error in prediction decreases (error curves become flatter) with increase in training samples used for training the neural network. The actual values of the prediction errors have been depicted in the last two rows of Column I, II, and III in Table 1.0. The maximum RMSE, MAE, and PMRE between the contaminant data computed by the ANN (ANN3) and the FD model when trained with optimal number of training samples are 0.0135, 0.0122, and 3.01%, respectively. Overall, the result based on ANN and the FD model show good agreement. Although the prediction errors gradually increase with increase in prediction times, they show decreasing trend when more samples are used for training the back propagation neural network model.

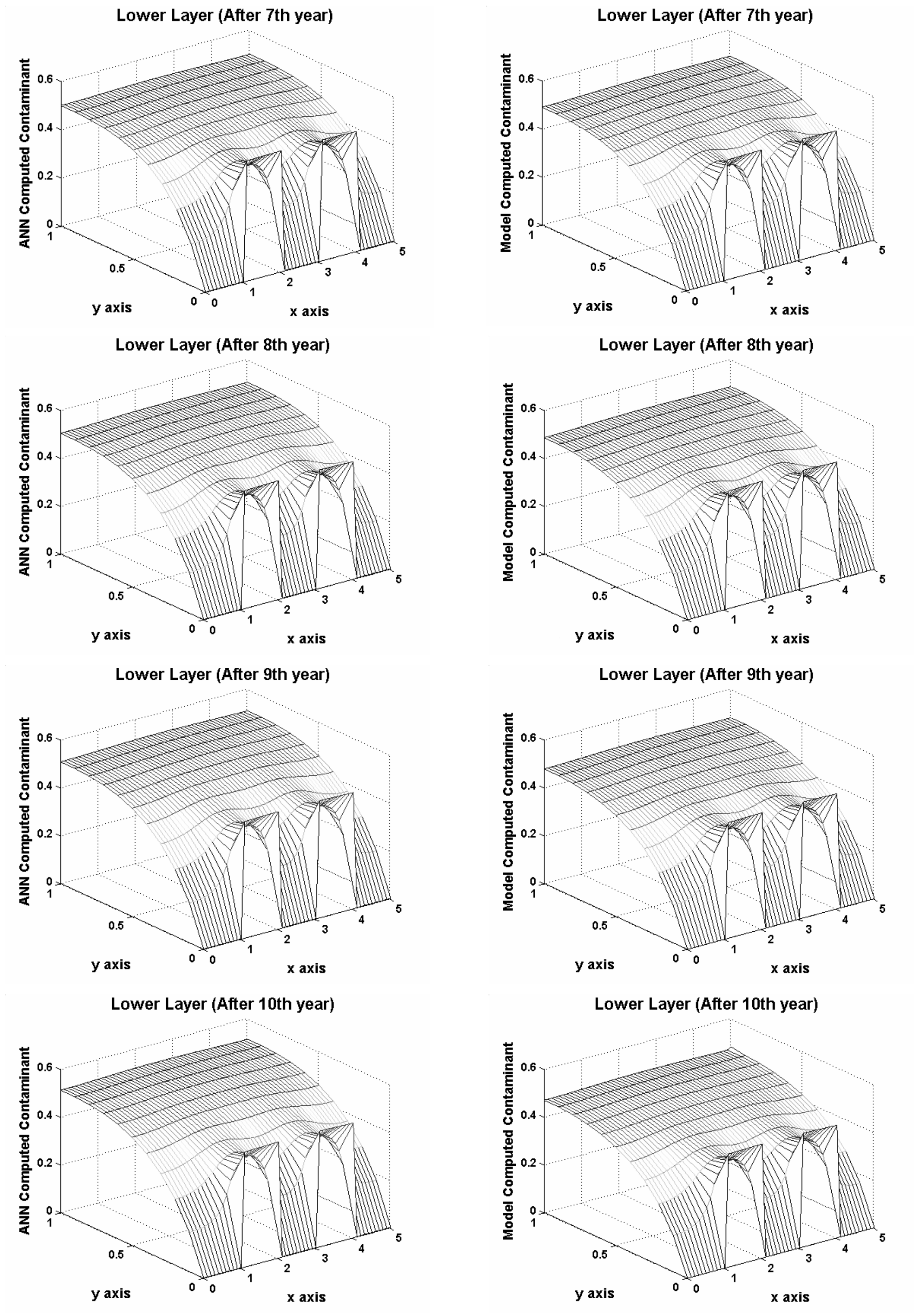


Figure 8 : Comparison of the Lower Layer Contaminant Values Computed by ANNI and the FD Model for Four Subsequent Years

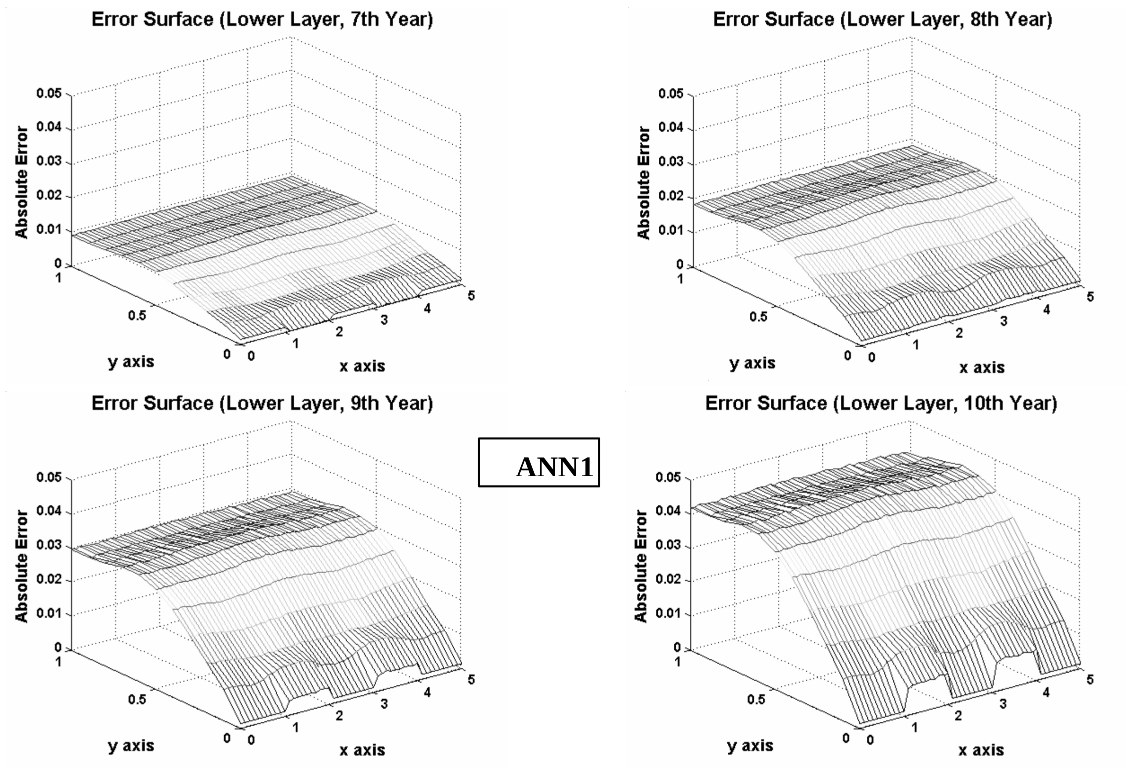


Figure 9 : Plot of the Absolute Error Between Contaminant Values Computed by ANNI and the FD Model for Four Subsequent Years (7th, 8th, 9th and 10th year)

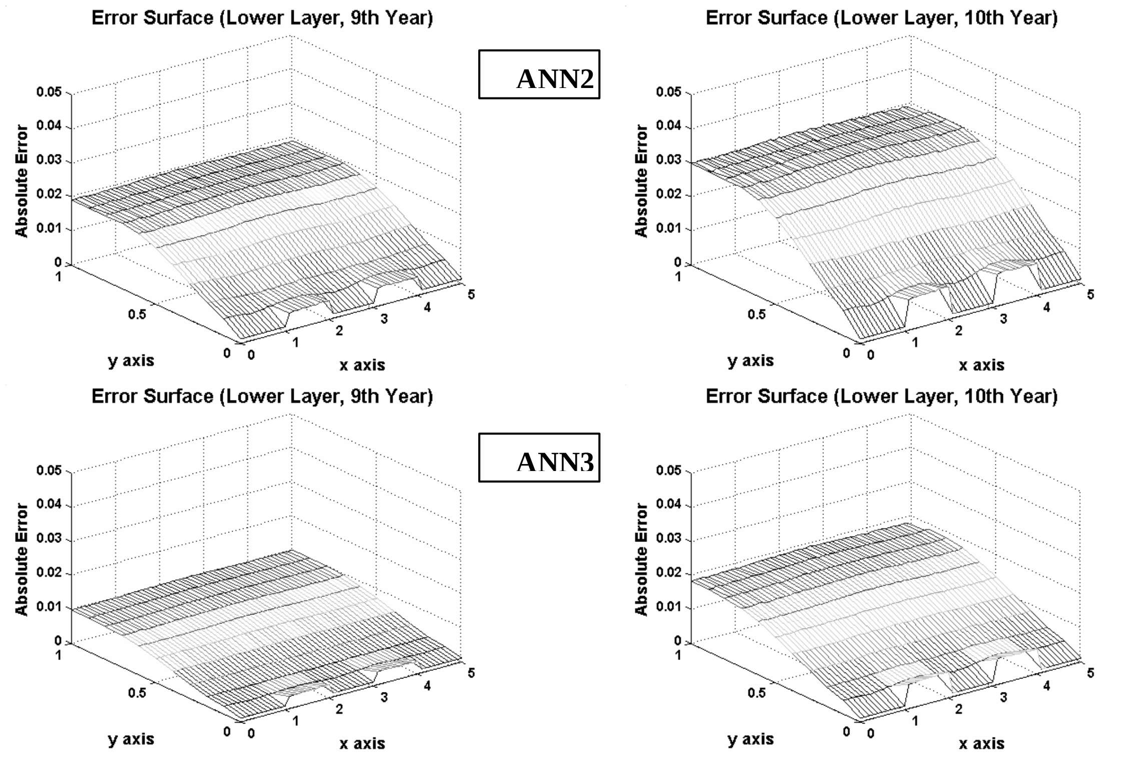


Figure 10 : Plot of the Absolute Error Between Contaminant Values Computed by ANN2 and FD Model (First two Plots) and ANN3 and FD Model (Last two Plots) for the 9th and 10th Year

6. CONCLUDING REMARKS

An ANN-based methodology to determine the contaminant values in the upper layer of the aquifer based on the values in the lower layer was presented and tested using the aquifer data taken over several years. The methodology is based on the idea that although the spread of contaminant values in the upper and lower layers of the aquifer changes over time due to various factors, the variation of contaminant values are interrelated and can be well estimated through ANN simulations. A three-layer feed-forward back propagation ANN with a nonlinear differentiable log-sigmoid transfer function in both the hidden and the output layer, and a variable learning rate has proved to be useful in replacing the traditional modeling of transport processes. In order to study the growth of error in prediction with increasing prediction times, the back propagation neural network model was separately trained with different sets of training data (comprising pairs of the upper and lower layer contaminant values) to obtain three separate trained networks. Each trained network was then tested with pairs of contaminant values for the remaining years. To measure the error between ANN predicted contaminant values and those computed by the Finite Difference model, we have used three different error measures, namely, the root mean square error, the mean absolute error, and the percent mean relative error, for proper interpretation of the results. Overall, the results based on ANN and the FD models have shown good agreement. Although the prediction errors gradually increase with increase in prediction times, they show declining trend when more samples are used for training the back propagation neural network model. The ANN based prediction versus the model-based observations of the contaminant values are characterized by a maximum root mean square error of 0.0135, maximum mean absolute error of 0.0122, and maximum percent mean relative error of 3.01%, when trained with optimal number of training samples. The application of this validated neural network methodology could be applied in other scenarios where complicated hydro-geologic parameters are involved while estimating aquifer pollution.

REFERENCES

[1] ASCE Task Committee on Application of Artificial Neural Networks in Hydrology, Artificial Neural Networks in Hydrology. II: Hydrologic Applications, Journal of Hydrologic Engineering, ASCE. **5** (2000) 124-137.

[2] J. Bear, A. Verrujit, Modeling Groundwater Flow and Pollution. D-Reidel Publishing Company; Netherlands, 1990.

[3] J. Carrera, State of the Art of the Inverse Problem Applied to the Flow and Solute Transport Equations. Groundwater Flow and Quality Modeling, D. Reidel Publishing Corporation, 1988.

[4] T. A. Clair, J. M. Ehrman, Variations in Discharge and Dissolved Organic Carbon and Nitrogen Export from Terrestrial Basins with Changes in Climate: A neural network approach, Limnology & Oceanography. **41** (1996) 921-927.

[5] P. Coulibaly, F. Anctil, R. Aravena, B. Bobee, Artificial Neural Network Modeling of Water Table Depth Fluctuations. Water Resources Research. **37** (2001) 885-896.

[6] S. K. Das, S. K. Roy, L. K. Ghosh, R. S. Kankara, Numerical Simulation of Solute Transport in Porous Media with First Order Chemical Reaction, *Indian Journal of Environmental Protection*. 20 (2000) 721-730.

[7] H. Demluth, M. Beale, Neural Network Toolbox for use with MATLAB, Users Guide Version 3, The MathWorks, Inc, Natic, Maine, 2000.

[8] R. Hechst-Nielsen, Neurocomputing, Addison-Wesley Publishing Company, 1990

[9] I. Knowles, Parameter Identification for Elliptic Problems, *Journal of Computational and Applied Mathematics.* **131** (2001) 175-194.

[10] H. Jeffreys, B. S. Jeffreys, The Gibbs Phenomenon, Methods of Mathematical Physics, Cambridge University Press, Cambridge, England, 1988.

[11] H. R. Maier, G. C. Dandy, Neural Networks for the Prediction and Forecasting Water Resources Variables: A Review of Modeling Issues and Applications. Environmental Modelling and Software. **15** (2000) 101-124.

[12] T. Masters, Practical Neural Network Recipes in C++, Academic Press, New York, 1993.

[13] B. S. Mazumdar, S. K. Das, Effect of Boundary Reaction on Solute Dispersion in Pulsatile Flow through a Tube, Journal of Fluid Mechanics. **239** (1992) 523-549.

[14] J. Morshed, J. J. Kaluarachchi, Application of Artificial Neural Network and Generic Algorithm in Flow and Transport Simulations. Advances in Water Resources. **22** (1998) 145-158.

[15] N. L Poff, S. Tokar, P. Johnson, Stream Hydrological and Ecological Responses to Climate change Assessed with an Artificial Neural Network, Limnology & Oceanography. **41** (1996) 857-863.

[16] L. L Rogers, F.U. Dowla, Optimization of Groundwater Remediation using Artificial Neural Networks with Parallel Solute Transport Modeling. Water Resources Research. **30** (1994) 457-481.

[17] L. L Rogers, F.U. Dowla, V. M. Johnson, Optimal field-scale Groundwater Remediation using Neural Networks and the Genetic Algorithm. Environmental Science and Technology. **29** (1995) 1145-1155.

[18] D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning Internal Representations by Error Propagation. In Parallel Distributed Processing, Vol. 1, ed. Rumelhart, D. E. and J. L. McClelland, 318-362. Cambridge, MA: MIT Press. 1986.

[19] P. K. Simpson, Artificial Neural Systems: Foundations, Paradigms, Applications, and Implementations, Pergamon Press Inc. 1990.

[20] J. Smith, R. N. Eli, Neural-network models of Rainfall-runoff Processes. Journal of Water Resources Planning and Management. **121** (1997) 499-508.

[21] A. S. Warke, S. K. Das, Modeling of Contaminant transport in a Layered Aquifer, Nature, Environment and Pollution Technology. **3** (2004) 421-428.

[22] M. L. Zhu, M. Fujita, N. Hashimoto, Application of Neural Networks to Runoff Prediction, in Stochastic and Statistical Methods in Hydrology and Environmental Engineering, Kluwer Academic Publishers, Mass. 1994.

[23] W.W-G. Yeh, Review of Parameter Identification Procedures in Groundwater Hydrology: The Inverse Problem, Water Resources Research. **22** (1986) 95-108.

[24] J. Scott Armstrong and Fred Collopy, Error Measures For Generalizing About Forecasting Methods: Empirical Comparisons. International Journal of Forecasting, 8 (1992), 69-80.

[25] C. Chatfield, Apples, Oranges and Mean Square error, International Journal of Forecasting, **4** (1988), 515-518.

[26] E. O. Frind. Simulation of long-term Transient Density-dependent Transport in Groundwater, Advances in Water Resources, **5** (1982), 73-88.

S.S. Kadam
Centre for Development of Advanced Computing (C-DAC),
Pune University Campus, Ganeshkhind, Pune- 411007, India
Email:sskadam@cdac.in

S.K. Das
International Institute of Information Technology (I^2IT),
P-14, Pune Infotech Park, Pune - 411057, India
Email: samirkumar_d@yahoo.com / samird@isquareit.ac.in

A.S. Warke
S.V.C. Polytechnic, Vadgaon b.k, Pune- 411 041, India
Email: krinda2003@yahoo.co.in