Sigma J Eng & Nat Sci 36 (1), 2018, 251-260

Sigma Journal of Engineering and Natural Sciences
Sigma Mühendislik ve Fen Bilimleri Dergisi

Research Article

IMPLEMENTATION OF AN MS EXCEL TOOL FOR BACKPROPAGATION NEURAL NETWORK ALGORITHM IN ENVIRONMENTAL ENGINEERING EDUCATION

Selami DEMİR*[1], Neslihan MANAV DEMİR[2], Aykut KARADENİZ[3], Hülya CİVELEK YÖRÜKLÜ[4]

[1] *Yıldız Technical University, Environmental Engineering Dep., ISTANBUL;* ORCID:0000-0002-8672-9817
[2] *Yıldız Technical University, Environmental Engineering Dep., ISTANBUL;* ORCID:0000-0002-6050-6308
[3] *Yıldız Technical University, Environmental Engineering Dep., ISTANBUL;* ORCID:0000-0002-9754-9088
[4] *Yıldız Technical University, Environmental Engineering Dep., ISTANBUL;* ORCID:0000-0002-3292-7084

Received: 06.04.2017 Revised: 25.12.2017 Accepted: 25.12.2017

ABSTRACT

This paper presents the implementation of an MS Excel tool for backpropagation neural networks in environmental engineering education. A number of test cases, Darcy friction factor and oxygen solubility, were also provided to test the performance of the tool. Relative mean square errors and coefficients of determination for Darcy friction factor were calculated as $1.53*10^{-3}\pm0.59*10^{-3}$ and 0.9983±0.0003 while they were calculated as $4.78*10^{-4}\pm2.33*10^{-4}$ and 0.9998±0.0000 for oxygen solubility. Results suggested that the tool's performance is satisfactory. The tool produces satisfactorily fast and reliable results and can be used for environmental education in undergraduate/graduate level.

Keywords: Environmental modeling, education, neural networks, visual basic for applications.

1. INTRODUCTION

Most studies in environmental modeling field involves developing differential equations as well as model formulations based on physical, chemical and biochemical phenomena, and the modeler needs to have fundamental knowledge on how to solve these formulations analytically or numerically, examples of which include modeling of activated sludge systems for wastewater treatment [1, 2], and source apportionement of several air pollutants [3, 4]. However, implementing and solving these formulations on computer requires advanced programming skills in addition to the fact that solution is, most of the time, time-consuming. Besides, solving these formulations requires a thorough understanding of complicated and nonlinear mathematical relationships behind the engineering system of interest. Although there usually exists commerical softwares for solving these kinds of phenomenological models, they are usually expensive and lead to increased costs of projects.

An artificial neural network (ANN) is a modeling tool for simulating engineering systems that involves complicated, nonlinear relationships between the system inputs and outputs without

[*] Corresponding Author: e-mail: seldemir@yildiz.edu.tr, tel: (212) 383 53 72

understanding the nature of the phenomena [5]. They are very similar to the human nervous system in that neurons can perform a given task by operating collectively [6]. Artificial neural networks can be classified with respect to their structure and learning mechanisms including generalized regression neural networks (GRNN), radial basis function neural networks (RBFNN), and backpropagation neural networks [7]. Of these BPNNs are most commonly used [8].

Artificial neural network neurons send signals to each other to establish communication for the ultimate goal of learning the complicated relationship between system inputs and outputs [9]. Neurons in an ANN are arranged in a manner that the user can select number of neurons in successive layers, and are connected to each other by linear combinations of weights. The learning process is combination of a forward run and a backward run for each data point in the training set. In the forward run, the neurons produce a signal and transmit its signal to the neurons in the preceding signal, successively until an estimate of the system output is made for the given inputs. The learning process take place in the backward run in which all weights are updated by backpropagating the discrepancy between the target and predicted values of system output. A set of forward and backward runs for each data point in the training set is called an epoch and a well-trained ANN requires a few thousand epochs in the learning process for well-established predictions of the phenomena involved.

Over years, artificial neural networks have been successfully employed for environmental modeling works which usually consist of systems with complicated, nonlinear relationships between inputs and outputs [5-13]. Motivation of the authors for implementing an open-source, user-friendly ANN software comes from the fact that ANNs prove to be excellent tools for simulating environmental systems and are employed in many types of environmental problems, that available commercial softwares for ANN lead to increased costs of research work, and that it can be used as an excellent modeling tool in environmental engineering education. In this regard, the aim of this study is to develop an open-source, user-friendly tool for artificial neural networks for use in undergraduate– and graduate–level education in environmental sciences. Since most of the PC owners also own at least a student-version of Microsoft Excel, Microsoft Visual Basic for Applications (VBA) was selected as the programming platform. This paper presents implementation of the neural network algorithm along with a number of modeling problems to test the performance of the implemented software.

2. MATERIALS AND METHODS

2.1. Neural Network Algorithm

Fig. 1 shows general structure of a multilayer neural network in which l layers exists. The first and the last layers (1st and lth layers) are input and output layers. Each layer has a number of neurons n^i (n^1, n^2, ... n^{l-1}, n^l, respectively for each layer). These neurons are connected to each other by weights $w^i(j, k)$. Besides, each neuron (except those in the input layer) has a bias $b^i(j)$.

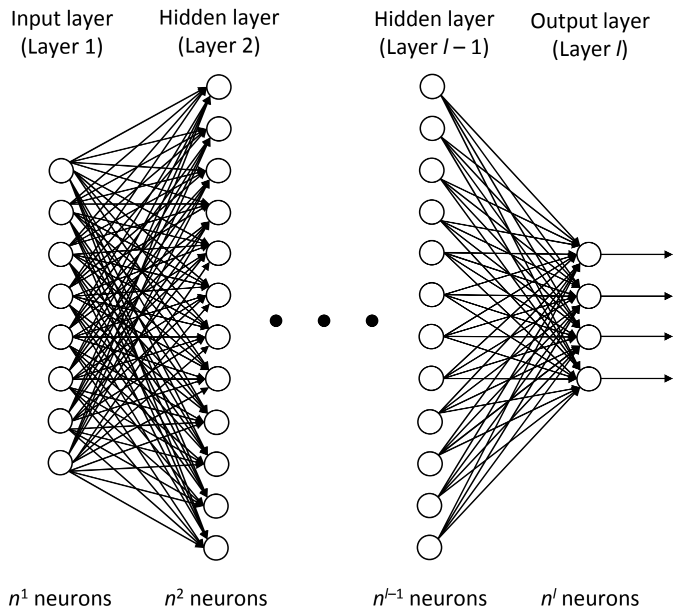


Figure 1. General structure of an artificial neural network

A backpropagation neural network starts with selecting a set of training examples from the data set which consists of n^1 independent variables (x) and n^l dependent variables (y). Note that the numbers of independent and dependent variables are equal to the numbers of input and output neurons, respectively. The number of data points in the data set is m. The set of training examples is a fraction of the data set, which is usually taken as 0.60 to 0.80.

The BPNN algorithm requires forward– and backward– runs for each training example during the learning process. The forward run starts with setting the activation for the input layer neurons a^1 to the values of the independent variables. From this point forward, this set of values is not updated during the learning process. In the forward run, the next step is to calculate the input sums (z^2) and activations (a^2) of all neurons in the second layer (l = 2). The calculation process is performed for each layer until the activations of the neurons in the output layer is finally computed. Following equations are used for this purpose:

$$z^i = w^i a^{i-1} + b^i \quad i = 2, 3, \ldots l \tag{1.a}$$

$$a^i = f\left(z^i\right) \quad i = 2, 3, \ldots l \tag{1.b}$$

where f is a vectorizing function, which, when applied to every element j of a vector v, produces $f(v)_j = f(v_j)$.

The activations of the neurons in the output layer are used to compute the cost function C using the following quadratic function:

$$C = \frac{1}{2}\sum_{j=1}^{n^l}\left(y_j - a_j^l\right) \tag{2}$$

The cost function is the main criteria in the learning process. It reflects total discrepancy between values of the dependent variable (y, also called target values) and the predicted values which is a^l. The output error δ^l is calculated as follows:

$$\delta^l = \nabla_a C \otimes f'(z^l) \tag{3}$$

where $\nabla_a C$ is the derivative of the cost function with respect to the activations of the neurons in the output layer, f' is derivative of the activation function with respect to its input, and $\otimes$ is the Hadamard product of two vectors. The output error is propagated back to the second layer of the network using the following formula:

$$\delta^i = \left\{\left[w^{i+1}\right]^T \delta^{i+1}\right\} \otimes f'(z^i) \quad i = l-1, l-2, \ldots 2 \tag{4}$$

In the backward run, the network calculates the error for each layer and updates the biases and weights using the gradient of the cost function:

$$\frac{\partial C}{\partial b_j^i} = \delta_j^i \quad i = l, l-1, \ldots 2 \quad j = 1, 2, \ldots n^i \tag{5.a}$$

$$\frac{\partial C}{\partial w_{jk}^i} = a_k^{i-1}\delta_j^i \quad i = l, l-1, \ldots 2 \quad j, k = 1, 2, \ldots n^i \; k = 1, 2, \ldots n^{i-1} \tag{5.b}$$

$$b_j^i = b_j^i - \eta \frac{\partial C}{\partial b_j^i} \quad i = l, l-1, \ldots 2 \quad j = 1, 2, \ldots n^i \tag{5.c}$$

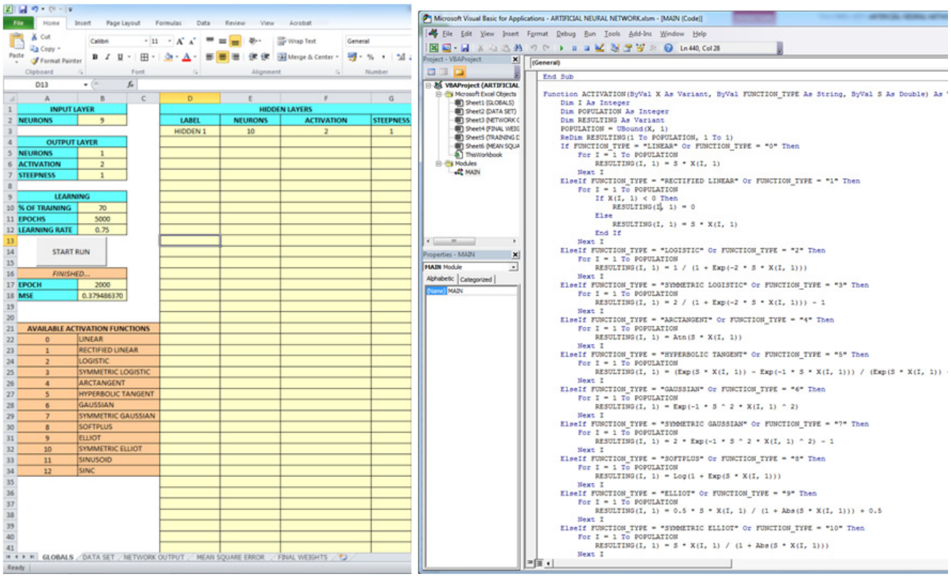
$$w_{jk}^i = w_{jk}^i + \eta \frac{\partial C}{\partial w_{jk}^i} \quad i = l, l-1, \ldots 2 \quad j, k = 1, 2, \ldots n^i \; k = 1, 2, \ldots n^{i-1} \tag{5.d}$$

where η is the learning rate. In the backward run, all of the weights and biases in the network are updated based on the error calculated between the target value and the predicted value for each dependent run. The complete set of forward and backward runs for each example in the training set is called an epoch. The network is trained for a predefined number of epochs until the learning process ends.

2.2. MS Excel

Visual Basic for Applications (VBA) is an event-driven programming language and is a derivative of Microsoft Visual Basic 6 (VB6). VBA is embedded in Microsoft Office software like Excel and is compiled to packed code (P-code). The P-code is then stored as a separate stream in an Excel file, independent of the spreadsheet stream. This property of VBA simplifies the problem of compiling the program into an executable file and of preparing a distribution package, if necessary, usually associated with stand-alone computer programming platforms, like VB6. Currently, VBA 7.1 is available through later versions Microsoft Office, however, there are still a great number of Microsoft Office 2010 users. Therefore, the codes and spreadhseets described in this paper is limited to MS Excel 2010 along with VBA 7.0.

An Excel VBA program was implemented that employs the backpropagation algorithm given in previous section. A screenshot of the Excel VBA program is shown in Fig. 2a. The software allows the user select the number of input and output neurons, the number of hidden layers, the activation function of hidden layers and output layer, the steepness coefficients for the activation functions, percent of training data, learning rate, and the number of epochs. Thirteen activation functions were embedded in the software along with their derivatives, namely, linear, rectified linear, logistic, symmetric logistic, arctangent, hyperbolic tangent, Gaussian, symmetric Gaussian, softplus, Elliot, symmetric Elliot, sinusoid, and sinc functions [14]. The software was prepared in modular structure, in which the user can go through the code to add new activation functions (Fig. 2b).



INPUT LAYER	
NEURONS	9

OUTPUT LAYER	
NEURONS	1
ACTIVATION	2
STEEPNESS	1

LEARNING	
% OF TRAINING	70
EPOCHS	5000
LEARNING RATE	0.75

START RUN

FINISHED...	
EPOCH	2000
MSE	0.379486370

AVAILABLE ACTIVATION FUNCTIONS	
0	LINEAR
1	RECTIFIED LINEAR
2	LOGISTIC
3	SYMMETRIC LOGISTIC
4	ARCTANGENT
5	HYPERBOLIC TANGENT
6	GAUSSIAN
7	SYMMETRIC GAUSSIAN
8	SOFTPLUS
9	ELLIOT
10	SYMMETRIC ELLIOT
11	SINUSOID
12	SINC

HIDDEN LAYERS			
LABEL	NEURONS	ACTIVATION	STEEPNESS
HIDDEN 1	10	2	1

GLOBALS / DATA SET / NETWORK OUTPUT / MEAN SQUARE ERROR / FINAL WEIGHTS

Microsoft Visual Basic for Applications - ARTIFICIAL NEURAL NETWORK.xlsm - [MAIN (Code)]

```
End Sub

Function ACTIVATION(ByVal X As Variant, ByVal FUNCTION_TYPE As String, ByVal S As Double) As
    Dim I As Integer
    Dim POPULATION As Integer
    Dim RESULTING As Variant
    POPULATION = UBound(X, 1)
    ReDim RESULTING(1 To POPULATION, 1 To 1)
    If FUNCTION_TYPE = "LINEAR" Or FUNCTION_TYPE = "0" Then
        For I = 1 To POPULATION
            RESULTING(I, 1) = S * X(I, 1)
        Next I
    ElseIf FUNCTION_TYPE = "RECTIFIED LINEAR" Or FUNCTION_TYPE = "1" Then
        For I = 1 To POPULATION
            If X(I, 1) < 0 Then
                RESULTING(I, 1) = 0
            Else
                RESULTING(I, 1) = S * X(I, 1)
            End If
        Next I
    ElseIf FUNCTION_TYPE = "LOGISTIC" Or FUNCTION_TYPE = "2" Then
        For I = 1 To POPULATION
            RESULTING(I, 1) = 1 / (1 + Exp(-2 * S * X(I, 1)))
        Next I
    ElseIf FUNCTION_TYPE = "SYMMETRIC LOGISTIC" Or FUNCTION_TYPE = "3" Then
        For I = 1 To POPULATION
            RESULTING(I, 1) = 2 / (1 + Exp(-2 * S * X(I, 1))) - 1
        Next I
    ElseIf FUNCTION_TYPE = "ARCTANGENT" Or FUNCTION_TYPE = "4" Then
        For I = 1 To POPULATION
            RESULTING(I, 1) = Atn(S * X(I, 1))
        Next I
    ElseIf FUNCTION_TYPE = "HYPERBOLIC TANGENT" Or FUNCTION_TYPE = "5" Then
        For I = 1 To POPULATION
            RESULTING(I, 1) = (Exp(S * X(I, 1)) - Exp(-1 * S * X(I, 1))) / (Exp(S * X(I, 1))
        Next I
    ElseIf FUNCTION_TYPE = "GAUSSIAN" Or FUNCTION_TYPE = "6" Then
        For I = 1 To POPULATION
            RESULTING(I, 1) = Exp(-1 * S ^ 2 * X(I, 1) ^ 2)
        Next I
    ElseIf FUNCTION_TYPE = "SYMMETRIC GAUSSIAN" Or FUNCTION_TYPE = "7" Then
        For I = 1 To POPULATION
            RESULTING(I, 1) = 2 * Exp(-1 * S ^ 2 * X(I, 1) ^ 2) - 1
        Next I
    ElseIf FUNCTION_TYPE = "SOFTPLUS" Or FUNCTION_TYPE = "8" Then
        For I = 1 To POPULATION
            RESULTING(I, 1) = Log(1 + Exp(S * X(I, 1)))
        Next I
    ElseIf FUNCTION_TYPE = "ELLIOT" Or FUNCTION_TYPE = "9" Then
        For I = 1 To POPULATION
            RESULTING(I, 1) = 0.5 * S * X(I, 1) / (1 + Abs(S * X(I, 1))) + 0.5
        Next I
    ElseIf FUNCTION_TYPE = "SYMMETRIC ELLIOT" Or FUNCTION_TYPE = "10" Then
        For I = 1 To POPULATION
            RESULTING(I, 1) = S * X(I, 1) / (1 + Abs(S * X(I, 1)))
        Next I
```

(a) (b)

Figure 2. Screenshots of the Excel VBA program. **a.** sheet view, **b.** code view

2.3. Test Case: Darcy Friction Factor

Fast and accurate calculation of Darcy friction factor is of importance for environmental engineers when dealing with large piping systems such as water distribution networks since it takes long times to complete calculations because of iterative scheme of solution. Colebrook [15] proposed an implicit formulation as an accurate alternative to well-known Moody's diagram for calculating Darcy friction factor. With the development of computers, a great number of papers have been dedicated to formulating explicit correlations of friction factor for computerized applications including [16-20], each of which provides accuracy and calculation speed to certain degrees.

In this study, 375 data points of Darcy friction factor by implicit Colebrook equation were used to test the performance of the implemented BPNN tool. The relative roughness (*RR*) and Reynolds numbers (*RE*) were between $1*10^{-6}$ to 0.05, and $4*10^{3}$ to $1*10^{8}$, respectively. Since friction factor is traditionally expressed as logarithmic functions of *RR* and *RE*, natural logarithms of *RR* and *RE* were used as the inputs to estimate the friction factors (*f*).

2.4. Test Case: Oxygen Solubility

Solubility of oxygen in water (*C*) that is in contact with air at a given atmospheric pressure and temperature governs the rate at which water gains oxygen. Therefore, it determines the capacity of a neutral water body to aerobically neutralize organic pollutants and is an important parameter for environmental engineers to assess water quality. For example, dissolved oxygen concentration needs to be kept at a sufficient level in order to properly operate of aerobic processes [21]. On the other hand, under anaerobic conditions, an increase in dissolved oxygen concentration negatively affects the efficiency of the treatment [22].

Solubility of oxygen is mainly a function of water temperature (*T*) and salinity (*S*) at a given atmospheric pressure. Although salinity is of minor concern for fresh water bodies compared to temperature, in some cases it becomes the major concern considering the fact that wastewater streams eventually find their way to seas. In this study, solubilities of oxygen in fresh and salt water was used to test performance of the implemented BPNN tool. For this purpose, oxygen solubilities at various water temperatures (between 0 and 45°C by 1°C) and various salinities (0.0, 9.0, 18.1, 27.1, 36.2 and 45.2 ppt) were used. A total of 276 data points were employed to train the BPNN tool.

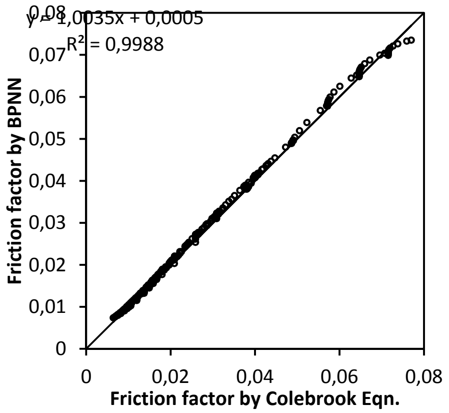
3. RESULTS

An MS Excel tool incorporating Visual Basic for Applications (VBA) was implemented for backpropagation neural network (BPNN) for undergraduate/graduate education in environmental engineering. The performance of the tool was tested using a couple of test cases, namely darcy friction factor and oxygen solubility in water. The results from the test cases are summarized in the following sections.

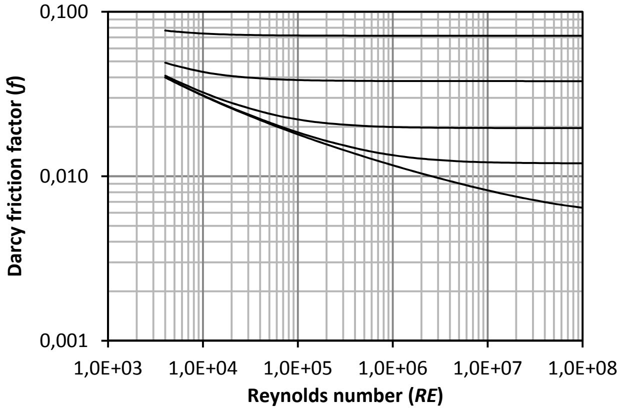
3.1. Darcy Friction Factor

Darcy friction factors were calculated using the implicit Colebrook equation for relative roughness $1*10^{-6} \leq RR \leq 0.05$, and Reynolds number $4*10^{3} \leq \mathrm{RE} \leq 1*10^{8}$. A total of 375 data points were used. Natural logarithms of *RR* and RE^{-1} ($ln[RR]$ and $ln[RE^{-1}]$) were used as inputs and *f* values by Colebrook equation were used as target values. Randomly selected 70% of data were used for training. The learning rate was 0.75 and the number of epochs was 5000. Ten neurons were employed in the single hidden layer with logistic function as the activation function. Since the BPNN tool randomly selects the training set, the tool was run for 25 times to ensure the convergence to the global minimum of the cost function.

The highest and the lowest values of relative mean square errors (RMSE) from 25 runs were calculated as $1.05*10^{-3}$ and $2.98*10^{-3}$, respectively, with an average of $1.53*10^{-3}\pm0.59*10^{-3}$. The calculated coefficients of determination were between 0.9977 and 0.9988 with an average value of 0.9983±0.0003. Fig. 3.a shows correlation plot between Colebrook- and BPNN-friction factors for best BPNN performance. The slope and intercept of a regression line between calculated values of friction factor were predicted as 1.0035 and 0.0005 showing that the BPNN results agree well with the Colebrook values. Another VBA code was also written to calculate friction factors by BPNN with the trained-weights and the calculated values of friction factors are shown in Fig. 3.b. The speed of calculation of friction factor is another concern when dealing with large pipe systems. Darcy friction factor by BPNN was calculated 10^{6} times to test the speed of BPNN prediction method. The speed of BPNN prediction was satisfactory (calculations were completed in around 15 seconds on a computer Intel Core i5-2500K CPU on 3.3GHz and 16 GB RAM at 2133 MHz).



(a)



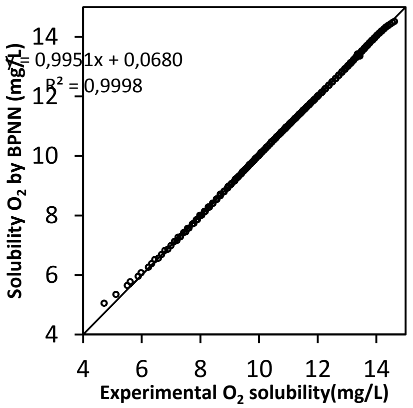
(b)

Figure 3. BPNN results. **a.** correlation plot between calculated friction factors by Colebrook equation and by BPNN, **b.** Calculated friction factors by BPNN as a function of *RR* and *RE*.

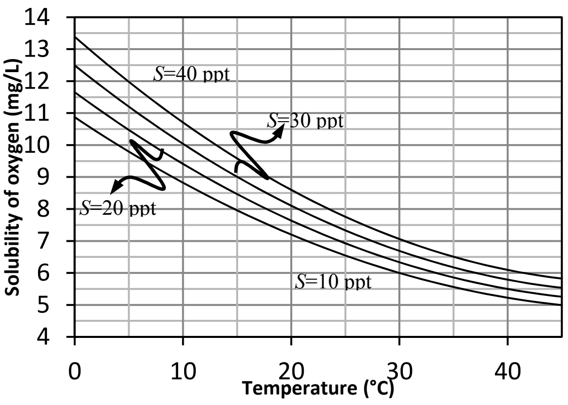
3.2. Oxygen Solubility

Solubility of oxygen at water temperatures (*T*) between 0 and 45°C and salinities (*S*) between 0 and 45.1 ppt were used for training BPNN tool. The neural network consisted of one input layer with two neurons (*T* and *S*), one hidden layers with ten neurons, and one output layer with single neuron (*C*). Sinc function was used as the activation function. The number of epochs was set to 5000 and the learning rate was 0.75. Randomly selected 70% of 276 data points were used for training. The tool was run for 25 times to ensure the convergence to the global minimum of the cost function.

The highest and the lowest values of relative mean square error (RMSE) between target and predicted values of oxygen solubility were calculated as $1.52*10^{-4}$ and $9.06*10^{-4}$, respectively. Average RMSE from 25 runs was $4.78*10^{-4}\pm2.33*10^{-4}$. Average coefficient of determination from 25 runs was calculated as 0.9998±0.0000. Experimental and predicted values of oxygen solubility are shown in Fig. 4. Fig. 4.a is a correlation plot between experimental and predicted values. The slope and intercept of the regression line were calculated as 0.9951 and 0.0680, respectively, which, along with satisfactorily low RMSE values, indicates that BPNN-predicted values agree well with the experimental values. Solubilities of oxygen at salinities other than those employed in training set were calculated using the trained BPNN and is shown as a function of temperature and salinity in Fig. 4.b. Smooth curves of oxygen solubilities supports the predicted values.



(a)



(b)

Figure 4. BPNN results. **a.** correlation plot between experimental and BPNN solubilities of oxygen, **b.** Calculated solubilities of oxygen by BPNN as a function of temperature and salinity.

4. CONCLUSIONS

An MS Excel tool incorporating Visual Basic for Applications (VBA) for backpropagation neural network was engineered for undergraduate/graduate-level education in environmental engineering field. The tool was prepared as open-source and distributed to students via internet. The tool solves a backpropagation neural network training problem by minimizing sum of square of differences between target and predicted values of a dependent variable. The implemented tool has a user-friendly interface and allows the user-inputs of the number of neurons in input, hidden, and output layers, the activation functions, the steepness coefficient for activation functions, the number of epochs, the learning rate, and the number of hidden layers. Currently, a total of 13 activation functions are available within the tool. Since the tool is an open-source software, the user can define new activation functions as desired. Besides, the tool uses a user-defined percentage of the input data in training. After the training is over, the tool highlights the data points that are used in the training procedure.

In undergraduate/graduate-level studies in environmental engineering, students often need a readily-available tool for making quick estimations of design parameters for engineering systems. The tool covers this need.

An "environmental modeling" course (undergraduate) and a "prediction models in environmental engineering" course (graduate) are offered in Yıldız Technical University (Istanbul, Turkey). The implemented MS Excel VBA tool is the backbone of computerized education strategy in these courses.

ABBREVIATIONS

l	number of layers in the network
n^i	number of neurons in the i^{th} layer
$w^i(j, k)$	the weight matrix that connects k^{th} neuron in $(i-1)^{th}$ layer to j^{th} neuron in i^{th} layer
b^i	vector of biases in i^{th} layer
m	number of data points in the data set
x	vector of independent variables
y	vector of dependent variables
a^i	vector of activations in i^{th} layer
z^i	vector of input sums in i^{th} layer
δ^i	vector of error in i^{th} layer
η	learning rate

REFERENCES

[1] Henze M., Gujer W., Mino T., Matsuo T., Wentzel M.C., Marais G.v.R., (1999) Activated sludge model No. 2d, *Water Sci. Technol.* 39(1), 165-182.

[2] Ostace G.S., Critea V.M., Agachi P.Ș., (2011) Cost reduction of the wastewater treatment plant operation by MPC based on modified ASM1 with two-step nitrification/denitrification model, *Comput. Chem. Eng.* 35, 2469-2479.

[3] Demir S., Saral A., (2011) A new modification to the chemical mass balance receptor model for volatile organic compound source apportionment, *Clean-Soil Air Water* 39(10), 891-899.

[4] Demir S., Saral A., Ertürk F., Kuzu S.L., Goncaloğlu B.İ., Demir G., (2012) Effect of diurnal changes in VOC source strengths on performances of receptor models, *Environ. Sci. Pollut. R.* 19(5), 1503-1514.

[5] Demir S., Karadeniz A., Manav-Demir N., (2016) Artificial neural network simulation of cyclone pressure drop: Selection of the best activation function in Iraq, *Pol. J. Environ. Stud.* 25(5), 1891-1899.

[6] Şamlı R., Sivri N., Sevgen S., Kiremitci V.Z., (2014) Applying artificial neural networks for the estimation of chlorophyll-a concentrations along the Istanbul Coast, *Pol. J. Environ. Stud.* 23(4), 1281-1287.

[7] Elsayed K., Lacor C., (2012) Modeling and pareto optimization of gas cyclone separator performance using RBF type artificial neural networks and genetic algorithms, *Powder Technol.* 217, 84-99.

[8] Demir S., Karadeniz A., Manav-Demir N., (2016) Using steepness coefficient to improve artificial neural network performance for environmental modeling, *Pol. J. Environ. Stud.* 25(4), 1467-1477.

[9] Bayram A., Kankal M., (2015) Artificial neural network modeling of dissolved oxygen concentrations in a Turkish watershed, *Pol. J. Environ. Stud.* 24(4), 1507-1515.

[10] Jahandideh S., Jahandideh S., Asadabadi E.B., Askarian M., Movahedi M.M., Hosseini S., Jahandideh M., (2009) The use of artificial neural networks and multiple linear regression to predict rate of medical waste generation, *Waste Manage.* 29(11), 2874-2879.

[11] Aghav R.M., Kumar S., Mukherjee S.N., (2011) Artificial neural network modeling in competitive adsorption of phenol and resorcinol from water environment using some carbonaceous adsorbents, *J. Hazard. Mater.* 188, 67-77.

[12] Hernandez-Ramirez D.A., Herrera-Lopez E.J., Rivera A.L., del Real-Olvera J., (2014) Artificial neural network modeling of slaughterhouse wastewater removal of COD and TSS by electrocoagulation, *Stud. Fuzziness Soft Comput.* 312, 273-280.

[13] Ye J., Cong X., Zhang P., Zeng G., Hoffmann E., Wu Y., Zhang H., Fang W., (2015) Operational parameter impact and back propagation artificial neural network modeling for phosphate adsorption onto acid-activated neutralized red mud, *Journal of Molecular Liquids* 216, 35-41.

[14] Sibi P., Jones A.A., Siddarth P., (2013) Analysis of different activation functions using back propagation neural networks, *Journal of Theoretical and Applied Information Technology* 47(3), 1264-1268.

[15] Colebrook C.F., (1939) Turbulent flow in pipes, with particular reference to the transition region between the smooth and rough pipe laws, *Journal of the Institution of Civil Engineers* 11(4), 133-156.

[16] Swamee P.K., Jain A.K., (1976) Explicit equation for pipe flow problems, *J. Hydraul. Eng-ASCE* 102, 657-664.

[17] Buzelli D., (2008) Calculating friction in one step, *Mach. Des.* 80, 54-55.

[18] Avcı A., Karagoz I., (2009) A novel explicit equation for friction factor in smooth and rough pipes, *J. Fluid. Eng-T ASME* 131(6), 061203.

[19] Brkic D., (2011) An explicit approximation of the Colebrook equation for fluid flow friction factor, *J. Petrol. Sci. Eng.* 29, 1596-1602.

[20] Fang X., Xu Y., Zhou Z., (2011) New correlations of single-phase friction factor for turbulent pipe flow and evaluation of existing single-phase friction factor correlations, *Nucl. Eng. Des.* 241, 897-902.

[21] Holenda B., Domokos E., Redey A., Fazakas J., (2008) Dissolved oxygen control of the activated sludge wastewater treatment process using model predictive control, *Comput. Chem. Eng.* 32, 1270-1278.

[22] Botheju D., Lie B., Bakke R., (2009) Oxygen effects in anaerobic digestion, *Model Ident. Control* 30(4), 191-201.