7th International Conference on Hydroinformatics
HIC 2006, Nice, FRANCE

ASSESSMENT OF LONGITUDINAL DISPERSION COEFFICIENT BY MEANS OF DIFFERENT NEURAL NETWORKS

ADAM PIOTROWSKI, PAWEŁ M. ROWIŃSKI, JAROSŁAW J. NAPIÓRKOWSKI
Institute of Geophysics, Polish Academy of Sciences, Ks. Janusza 64, 01-452 Warsaw, Poland

Multi-Layer Perceptron, Fuzzy and Radial-Basis Function neural networks, Nearest Neighbour approach, linear regression, 2nd order curvilinear regression, and 'classical' empirical formulae have been applied for evaluation of longitudinal dispersion coefficient for a river reach. In general, results achieved by means of each type of neural networks outperforms these obtained by other techniques.

INTRODUCTION

Description of longitudinal dispersion in rivers is still at front of the research problems within environmental hydrologists. The simplest and most popular in practical approach is the model based on one-dimensional advection–dispersion equation. When assumption of almost uniform concentration distribution of admixture (C) in a river cross-section (A) is valid, what may be true in a far distance from the release point, and the channel geometry is relatively simple, this equation may be written in the form of

$$\frac{\partial C}{\partial t} + U\frac{\partial C}{\partial x} = \frac{1}{A}\frac{\partial}{\partial x}\left(AE_x \frac{\partial C}{\partial x} \right) \tag{1}$$

where U is cross-sectional averaged velocity, x is longitudinal axis and E_x is longitudinal dispersion coefficient.

When one dimensional advection-dispersion equation is applied to a natural stream, the model assumptions are usually not fully satisfied and, therefore, what is claimed to be measured value of dispersion coefficient is loaded with a significant error ([7] and [8]).

APPLIED METHODS AND DATABASE DESCRIPTION

To treat the Fickian model (1) as a predictive tool, one needs to know how to relate the usually unknown longitudinal dispersion coefficient to basic hydraulic and morphometric characteristics of the natural stream under consideration. Numerous empirical and semi-empirical formulae have been elaborated [9], [11]. When channel depth, H (m), channel width, B (m), cross-sectional averaged water velocity, U (m s^{-1}) and bulk shear velocity, U^* (m s^{-1}) are known, one may compute the longitudinal dispersion coefficient by means of

$$E_x = 5.915 \cdot H \cdot U \cdot \left(\frac{B}{H} \right)^{0.62} \cdot \left(\frac{U}{U^*} \right)^{1.428} \tag{2}$$

Another example is the formula proposed in [2]:

$$E_x = \frac{0.15 \cdot H \cdot U^*}{8 \cdot \left[0.145 + \frac{U}{3520 \cdot U^*} \cdot \left(\frac{B}{H} \right)^{1.38} \right]} \cdot \left(\frac{B}{H} \right)^{\frac{5}{3}} \cdot \left(\frac{U}{U^*} \right)^2 \tag{3}$$

Recently a Multi-Layer Perceptron neural network (MLP) has been proposed for dispersion assessment [5]. It was based on the following hydraulic input variables: *B*/*H*, *U/U*∗, and 3*UB*. Database was composed only of data collected in the US rivers. One should be aware that results of the modelling based on short-data sets can be contaminated with significant errors.

A bit different and more diverse data base, with 81 measurements was used in [7]. It was shown that when sinuosity index is considered as additional input, MLP network results improve significantly. The same data base is exploited in the present paper.

In this paper the evaluation of longitudinal dispersion coefficient by means of Radial-Basis Function neural networks (RBF), Fuzzy neural networks (FNN), Nearest Neighbour approach (NN), linear regression (LR), 2nd order curvilinear multiple regression (CR) and formulae proposed in [9] and [2] will be presented and compared with results obtained in [7] by means of MLP network. Two versions, namely V_1 and V_2 of each model will be verified. The input measurements in V_1 included *B*/*H*, *U/U*∗, 3*UB*, in V_2 additionally sinuosity index was considered. Because dispersion coefficients range from 1486.40 m^2/s to only 0.19 m^2/s, in the present paper logarithms of all variables were applied as model inputs and output. 81 cases were divided randomly into training (50) and validation (31) sets.

Stopping criteria of MLP and FNN training algorithms were link to the value of objective function *(J)* computed for validation data set

$$J(\mathbf{d}) = \frac{1}{n} \sum_{j=1}^{n} (\log E_x^e(\mathbf{d}) - \log E_x)^2 \tag{4}$$

J depends on the vector of parameters (**d**) to be optimized, different for each model. In the above equation E^e_x represents the modelled value of longitudinal dispersion, *n* is number of data in the training or validation sets. Selection of the best of trained models of each type was also performed due to validation set results.

All applied models are able to approximate the values of output variable y based on the set of input variables $x_1, x_2, \ldots, x_N$, hence $y = f(x_1, x_2, \ldots, x_N)$. In case of this paper $N = 3$ in V_1, $N = 4$ in V_2, and $y = E^e_x$,. The most popular Multi-Layer Perceptron networks comprise of several nodes arranged in input, hidden and output layers, as presented in Fig. 1. The number of input and output nodes is always equal to the number of input and output variables, whereas the number of hidden nodes should be evaluated experimentally, and it was 2 and 3 in V_1 and V_2, respectively. It is important to stress that, especially in short-data case, the number of parameters – connected with number of nodes – should be as small as possible, to assure the robustness of optimisation problem. Nodes in consecutive layers are connected via weights (w, and v) with thresholds (indexed with 0). Discussion about selecting the best activation functions may be found for example in [10]. A sigmoidal function as activation function in the hidden layer has been chosen in the present study

$$E_x^e = v_0 + \sum_{j=1}^{H} v_j \cdot (1 + \exp(-w_{0j} - \sum_{i=1}^{N} w_{ij} \cdot x_i))^{-1} \tag{5}$$

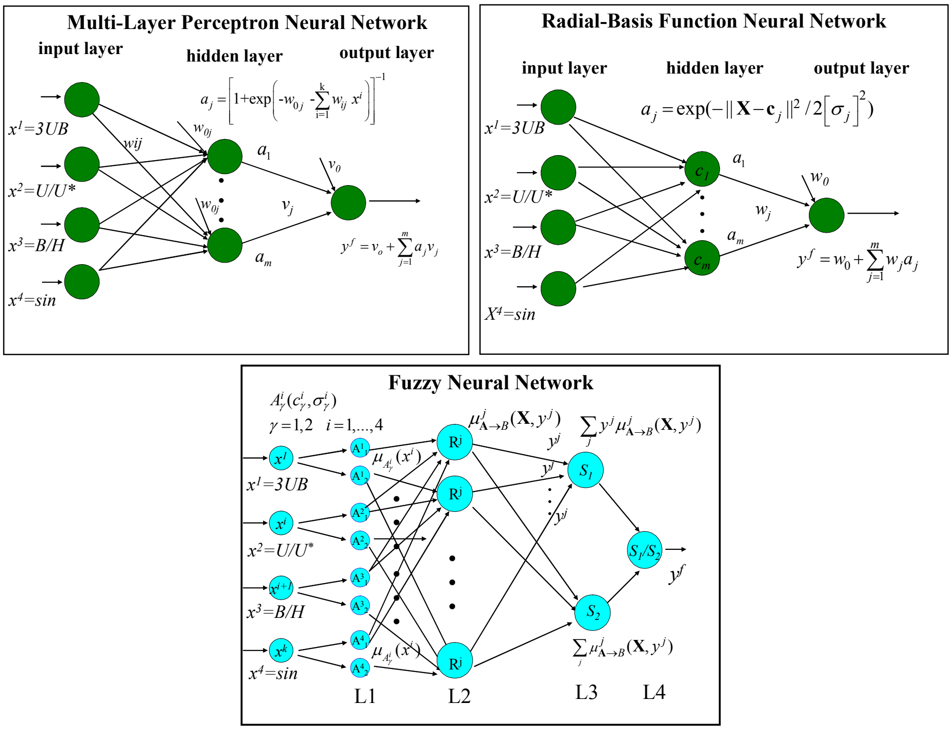
Multi-Layer Perceptron Neural Network
input layer
hidden layer
output layer
$a_j = \left[1+\exp\left(-w_{0j} - \sum_{i=1}^{k} w_{ij} x^i\right)\right]^{-1}$
$x^1=3UB$
$x^2=U/U^*$
$x^3=B/H$
$x^4=sin$
wij
w_{0j}
a_1
a_m
v_0
v_j
$y^f = v_o + \sum_{j=1}^{m} a_j v_j$
Radial-Basis Function Neural Network
input layer
hidden layer
output layer
$a_j = \exp(-\|\mathbf{X}-\mathbf{c}_j\|^2 / 2[\sigma_j]^2)$
$x^1=3UB$
$x^2=U/U^*$
$x^3=B/H$
$X^4=sin$
c_1
c_m
a_1
a_m
w_0
w_j
$y^f = w_0 + \sum_{j=1}^{m} w_j a_j$
Fuzzy Neural Network
$A^i_\gamma(c^i_\gamma, \sigma^i_\gamma)$
$\gamma=1,2 \quad i=1,\ldots,4$
$x^1=3UB$
$x^2=U/U^*$
$x^3=B/H$
$x^4=sin$
x^1 x^i x^{i+1} x^k
A^1_1 A^1_2 A^2_1 A^2_2 A^3_1 A^3_2 A^4_1 A^4_2
$\mu_{A^i_\gamma}(x^i)$
$\mu^j_{\mathbf{A}\to B}(\mathbf{X}, y^j)$
R^j
y^j
$\sum_j y^j \mu^j_{\mathbf{A}\to B}(\mathbf{X}, y^j)$
S_1
S_2
S_1/S_2
y^f
$\sum_j \mu^j_{\mathbf{A}\to B}(\mathbf{X}, y^j)$
L1
L2
L3
L4

Figure 1. Scheme of MLP, RBF and FNN models

For training the network, a Lavenberg-Marquardt nonlinear optimization algorithm [6] with multistart procedure has been applied, which should help to avoid sticking to local optima being far worse than the global one.

Radial-Basis Function neural networks differ from the MLP ones in the first two layers (Fig. 1), where a set of M radially symmetric, usually Gaussian functions with parameters **c** and σ, spanning N-dimensional space was applied (N being the number of inputs). M was experimentally set to 7 in both versions V_1 and V_2. When adding the weight parameters w from output layer, this model may be described in the form of

$$E_x^e = w_0 + \sum_{j=1}^{M} w_j \cdot \exp\left(-\frac{\left\| \mathbf{X} - \mathbf{c}_j \right\|_N^2}{2\sigma_j^2} \right) \tag{6}$$

where **X** is a vector of input variables. Spatial location of **c** was fixed by means of self-organized k-means algorithm. Deviations σ were evaluated separately for each of M parameters **c** by k-nearest neighbour rule. The number of nearest neighbours was experimentally set to 2 in V_1 and to 8 in V_2. The weights w in output layer require linear optimization when all other parameters are known – hence singular value decomposition algorithm was applied. Details of all these methods may be found in [4]. The advantage is that RBF network trained this way does not need to compute the value of objective function for the validation set as a stopping criteria.

The Fuzzy neural network models utilize a fuzzy logic theory [12], and were designed as a tool capable to deal with non-precise real-world data. Various types of FNN may differ significantly among one another due to abundance of possible fuzzification, defuzzification and inference methods which can be applied. Also the rule base may be composed in many ways. Two most popular types of models are Mammandi-Zadeh and Takagi-Sugeno ones [3]. In the present study FNN was constructed in the way depicted in Fig. 1.

Two fuzzy sets A^i_γ $(\gamma = 1,2)$ for each i-th input variable were considered in the fuzzification layer (L1), representing linguistic variables of "big" or "small". Membership functions (μ_A) of all sets were chosen to be Gaussian, i.e.

$$\mu_{A^i_\gamma}(x_i) = \exp\left[-\left(\frac{x_i - c^i_\gamma}{\sigma^i_\gamma} \right)^2 \right] \tag{7}$$

Parameters (c^i_γ, σ^i_γ), of each fuzzy set membership function were to be optimized, along with outputs of each rule (y^j) (see below) by back propagation algorithm with multistart procedure to allow avoiding local optima.). All possible fuzzy rules in the rule base layer (L2) are considered in the form of

$$R^j\text{: IF } (x_1 \text{ IS } A^1_\gamma \text{ AND ... AND } x_k \text{ IS } A^k_\gamma) \text{ THEN } (y^j \text{ IS } B^j) \tag{8}$$

Since k is the number of input variables (3 or 4 in version V_1 and V_2 respectively) the number of rules is γ^k, i.e. 8 for V_1 and 16 for V_2. Operation AND and THEN in Eq. (8) are represented by Cartesian product. Assuming that $\mu^j_B(y^j) = 1$ one may write

$$A^1_\gamma \times \ldots \times A^k_\gamma \times B^j = \mu_{A^1_\gamma}(x_1) \cdot \ldots \cdot \mu_{A^k_\gamma}(x_k) \cdot \mu^j_B(y^j) = \mu^j_{\mathbf{A}\to B}(\mathbf{X}, y^j) = \mu^j_{\mathbf{A}}(\mathbf{X}) \tag{9}$$

Finally, through defuzzification and inference layers (L3 and L4), the value of E^e_x is evaluated as

$$E^e_x = \frac{\sum_j y^j \mu^j_{\mathbf{A}}(\mathbf{X})}{\sum_j \mu^j_{\mathbf{A}}(\mathbf{X})} \tag{10}$$

The non-parametric Nearest Neighbour approach is exploited in pattern recognition, and in hydrology it was also applied to time series modelling [1]. This simple procedure may be also applied to evaluate longitudinal dispersion coefficient. Having k input variables, one may in k-dimensional space, for each input vector case $\mathbf{X}^j$, find its M nearest neighbours from training data set. As output values E_x are known for training data and for M nearest neighbours, one may evaluate the output value of $\mathbf{X}^j$ by, if $M > k$, applying linear regression method (see also below). The value of M should be chosen experimentally. When input vector case $\mathbf{X}^j$ belongs to training set, the point representing itself is not regarded as one of M nearest neighbours.

Linear regression model (LR) may be shortly described as

$$E^e_x = w_0 + \sum_{i=1}^{k} w_i x_i \tag{11}$$

whereas 2nd order curvilinear multiple regression (CR) is

$$E^e_x = w_0 + \sum_{i=1}^{k} w_i x_i + \sum_{j=1}^{k}\sum_{i=1}^{k} w_i x_i x_j \tag{12}$$

Both these approaches are also applied in this paper, along with empirical formulae of Eq. (2) and (3).

COMPUTATIONAL RESULTS

To compare the final results three indices were considered. The first one, Percentage of Mean Error (PME), following [5], has been evaluated as

$$PME = \frac{\frac{1}{n}\sum_{i=1}^{n}\log\left(\frac{E_{xi}^{e}}{E_{xi}}\right)}{\frac{1}{n}\sum_{i=1}^{n}\log E_{xi}}\cdot 100\% \tag{13}$$

Percentage index (PI) is

$$PI = \frac{\sum_{i=1}^{n} c_i}{n}\cdot 100\%, \quad c_i = \begin{cases} 1 \quad when \quad E_{xi}^{e} \in \left[\frac{E_{xi}}{A}, E_{xi}\cdot A\right] \\ 0 \quad when \quad E_{xi}^{e} \notin \left[\frac{E_{xi}}{A}, E_{xi}\cdot A\right] \end{cases} \tag{14}$$

where A is a determinative of range width. This way each case belonging to the range increases PI in the same degree, and may be treated as “hit” other as “missed”. Note that for different A, the results may differ significantly. One may also ask for the lowest A for which PI computed for particular model version (for training or validation set separately) is 100%. This value is called A-max. Third index, root mean square error (RMSE), is simply computed as root of (4) for proper data sets.

The results are presented in Table 1, for training and validation data sets separately. Empirical formulae could not be applied in V_2 case.

Similarly to [7] the results obtained from all models for V_2 are significantly better than for V_1. Formulae (2) and (3) fit the data even poorer than linear regression. It confirms the necessity to search for more reliable methods. Among other techniques all types of neural networks outperform LR, CR and NN methods in validation set cases, especially for V_2. It means that they are able to utilize additional information better than simpler approaches.

MLP networks seem to have more good “hits” than RBF and FNN ones (comparing PI results for A lower than 2). Note that for example for A = 1.2 FNN applied for validation set of V_1 data, “hits” only in 3% of cases (exactly – in one case over 31). On the other hand for A = 2 there is no difference between FNN and MLP, whereas for A = 3 FNN performs slightly better than MLP. But if comparing A-max values, FNN outperforms significantly all other models for V_1 validation set. Also for V_2 validation set, FNN – this time together with RBF model – has lower *A*-max values than MLP network. This may indicate that fuzzy logic allows to avoid the worst mistakes. It is important to note that – in validation set – the poorest "hits" are much better than poorest "hits" of MLP. But since FNN and RBF require much more parameters – in short-data modelling they are unable to compete in numbers of best "hits" for lower *A*. It is also seen when PME and RMSE are compared for validation data set of V_1, when the worst "misses" are not as significant as in V_2 and do not have as big influence on these statistics.

Table 1. Results summary

model	set	PME (%)	PME (%)	PI (%)					PI (%)					RMSE	RMSE
		V_2	V_1	V_2					V_1					V_2	V_1
				A											
				1.2	1.5	2	3	max	1.2	1.5	2	3	max		
MLP	training	7.12	13.97	50	86	92	96	4.8	26	50	72	94	5.9	0.168	0.275
RBF		9.60	16.30	38	68	84	98	4.5	8	36	68	92	3.6	0.204	0.286
FNN		10.03	16.14	46	68	86	96	6.0	22	40	62	88	5.5	0.227	0.306
NN		12.98	18.12	28	56	78	92	7.4	14	38	58	84	8.3	0.269	0.337
LR		12.05	17.39	26	58	82	96	5.7	18	28	62	90	6.2	0.239	0.312
CR		9.25	13.81	38	72	86	98	3.9	30	56	68	94	6.2	0.192	0.271
Seo and Cheong			22.71						16	32	54	70	10.2		0.443
Deng et al.			19.41						18	40	56	80	8.4		0.376
MLP	validation	10.27	28.02	45	77	90	94	4.0	23	35	45	65	22.8	0.196	0.485
RBF		12.13	25.70	39	58	84	97	3.1	29	39	48	74	28.8	0.205	0.464
FNN		13.43	27.60	23	68	77	97	3.4	3	26	48	68	7.8	0.218	0.415
NN		13.56	28.86	39	65	81	94	5.8	10	29	42	71	33.5	0.245	0.484
LR		17.34	31.53	16	55	68	94	5.4	13	23	39	61	33.8	0.279	0.524
CR		17.39	28.84	32	55	84	87	14.4	16	42	48	74	27.1	0.330	0.506
Seo and Cheong			39.37						16	23	42	55	44.3		0.659
Deng et al.			35.75						10	26	42	58	21.9		0.588

NN approach, although performs better than regression methods for validation sets, provides worse results than the ones obtained from neural networks.

CONCLUSIONS

Results obtained by means of each neural network type are found to be better than by empirical formulae (2) and (3), regression methods and Nearest Neighbours approach. These methods allow for the prediction of longitudinal dispersion coefficient and may ease the prediction of the pattern of pollution spread in rivers. The performance of all investigated methods improves significantly when sinuosity index is included. Although Multi-Layer Perceptron neural networks allow more precise evaluation of longitudinal dispersion coefficient than Radial-Basis Function or Fuzzy models, they are also more subjected to significant errors. Fuzzy models seem to evade the largest mistakes, which happened to other techniques, but, probably due to much more optimized parameters, they are unable to make comparatively many proper assessments as Multi-Layer Perceptron models. Radial-Basis Function neural networks also suffer overparametrization, although perform correctly, whereas Nearest Neighbour non-parametric methods rather yield to neural networks.

ACKNOWLEDGEMENT

This study was supported by the Polish Ministry of Education & Science – grants no 2P04D02629 and 2P04D00829.

REFERENCES

[1] Brath A., Montanari A. and Toth E., "Neural networks and non-parametric methods for improving real-time flood forecasting through conceptual hydrological models", *Hydrology and Earth System Sciences* 6(4), (2002), pp. 627-640.

[2] Deng Z. Q., Singh V. P. and Bengtsson L., "Longitudinal dispersion coefficient in straight rivers", *J. Hydraul. Engng ASCE* 127(11), (2001), pp. 919–927.

[3] Driankov D., Hellendoorn H. and Reinfrank M., "*An Introduction to fuzzy control*", Springer, Berlin, (1993).

[4] Haykin S., "*Neural Network, a Comprehensive Foundation*". Macmillan College Publishing Co., New York, USA, (1994).

[5] Kashefipour S. M., Falconer R. A. and Lin B., "Modeling longitudinal dispersion in natural channel flows using ANNs", in: *River Flow 2002* (ed. By D. Bousmar & Y. Zech)., A.A, Balkema/Swets & Zeitlinger, Lisse, The Netherlands, (2002), pp. 111–116.

[6] Press W. H., Flannery B. P., Teukolsky S. A. and Vetterling W. T., "*Numerical Recipes in C. The Art of Scientific Computing*", Cambridge University Press. Cambridge, UK, (1990).

[7] Rowinski P. M., Piotrowski A. and Napiórkowski J. J., "Are artificial network techniques relevant for the estimation of longitudinal dispersion coefficient in rivers?", *Hydrological Sciences Journal*, 50(1), (2005), pp. 175-187.

[8] Rutherford C. J., "*River mixing*", John Wiley & Sons, Chichester, UK, (1994).

[9] Seo I. W. and Cheong T. S, "Predicting longitudinal dispersion coefficient in natural streams", *Journal of Hydraulic Engineering* 124(1), (1998), pp. 25-32.

[10] Shamseldin A. Y., Nasr A. E. and O'Connor K. M., "Comparison of different forms of the Multi-layer Feed-Forward Neural Network method used for river flow forecasting", *Hydrology and Earth System Sciences*, 6(4), (2002), pp. 671-684.

[11] Sukhodolov A. N., Nikora V. I., Rowiński P. M., and Czernuszenko W., "A case study of longitudinal dispersion in small lowland rivers", *Water Environ. Res.* 69(7), (1997), pp. 1246–1253.

[12] Zadeh L. A., "Fuzzy sets", *Information and Control* 8 , (1965), pp. 338-353.