BULETINUL INSTITUTULUI POLITEHNIC DIN IAŞI
Publicat de
Universitatea Tehnică „Gheorghe Asachi” din Iaşi
Tomul LVIII (LXII), Fasc. 4, 2012
Secţia
CONSTRUCŢII. ARHITECTURĂ

PREDICTION OF DRILLING TORQUE FOR FULL DISPLACEMENT PILE CONSTRUCTION IN LOOSE SOIL USING NEURAL NETWORKS

BY

ISMAIL ABDUSSAMAD[1], IOAN-COSMIN SCURTU[2,*] and RALUCA-MARIA MIHALACHE[3]

[1]University of Dundee, Division of Civil Engineering, Scotland, UK
[2]"Gheorghe Asachi" Technical University of Iaşi
Faculty of Civil Engineering and Building Services
[3]"Gheorghe Asachi" Technical University of Iaşi
Faculty of Hydrotechnics, Geodesy and Environmental Engineering

Received: September 5, 2012
Accepted for publication: September 29, 2012

Abstract. The paper is aimed at developing a neural network model for predicting the drilling torque necessary to push a drill bit in to the subsurface during the installation of full displacement pile (FDP) based on the cone resistance value of the subsurface. A self-evolving network algorithm is employed developing the model in order to reduce the model complexity without compromising accuracy. The database used to construct and validate the model consists of measurements taken during a drilling operation to install FDP piles for a sedimentation tank of the Brăila Waste Water Treatment Plant. The results indicate that the neural network model predictions of drilling torque are in good agreement with filed observations.

Key words: FDP; drilling parameters; cone penetration test; neural network.

[*]Corresponding author: *e-mail*: scurtu.cosmin@gmail.com

1. Introduction

The installation process of Full Displacement Piles (FDP) is a relatively new technology used for bored cast *in situ* pile construction. Technologically they are the next step after the Continuous Flight Auger (CFA) type piles, and became feasible once the drilling rigs turned out to be powerful enough to provide the adequate torque needed to fully displace the soil. Even so, the applicability of these type of piles is limited to modest diameters, up to 610 mm, and to loose soil conditions.

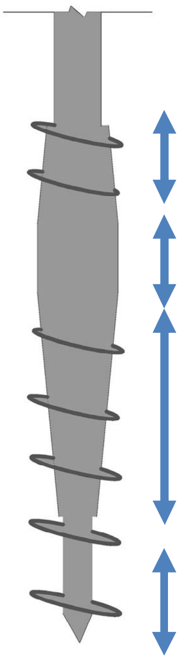
The data that is used for interpretation here was obtained at a site located in Brăila County, Romania. The site is characterized by a relatively uniform soil profile over a large area. This is confirmed by the cone penetration tests (CPT) done over a 400 m radius area, which gave quite similar results. The soil conditions are poor, the soil profile being made up of a lightweight loess deposit roughly 20 m thick, with dense sand lenses. Beneath the loess lies a deposit of cemented gravel. This profile makes this location ideally suited for the use of FDP. The piles were used as rigid intrusions, in order to improve the soil bearing capacity, so it will be possible the use of strip foundations. The drilling rigs provide the means of monitoring several key parameters during the making of the pile void and during the concreting of the pile stages. The applied force, the drilling torque, the rate of progress, the concrete pressure and the working pressure of the hydraulic pump are usually monitored. These parameters vary widely depending on the nature of the displaced soil. Some researchers have tried to figure out way of linking these parameters (in the case of bored piles) to the geotechnical characteristics of the soil that was being drilled into (Basu & Prezzi, 2009).

In the case of FDP it becomes of much interest the ability of one to estimate the amount of energy needed to fully displace the soil in order to produce the hole for the pile. As it will be demonstrated the amount of energy used by the pilling rig is in close connection with the amount of momentum needed to spin the displacement bit. So it is easy to notice that once a relation between the nature of the soil subject to drilling (as identified from CPT tests) and the torque needed for the advancement of the displacement bit, an estimate of the energy input can be made and also some other parameters can be derived (*i.e.* maximum torque for a given advance rate and so on).

In what follows, a self-evolving neural network is used to estimate the torque required to penetrate through the soil at certain rotational speed and rate of penetration, given the soil resistance, which is represented by CPT values. A field data obtained from the piling operations in the aforementioned site is used to train and test the proposed neural network model. In order to use a minimum possible network size for the modeling, a self-evolving neural network is used, where both network parameters and structure are simultaneously optimized during the training process. As a means of assessing the quality of its prediction, the performance of the proposed model is compared with conventional back propagation network (BPN) and general regression neural network (GRNN).

2. Full Displacement Pilling

Full displacement piling offers a relatively quiet and less chaotic alternative to driven piles. The vibration free nature of the installation operation makes it particularly suitable for pile construction in well developed urban areas. As in the case of driven piles, no soil material is removed from the earth during the installation of full displacement piles, which leads to an increase in shaft friction and end bearing capacity as well as minimizing waste generation in the process. Construction of FDPs involves using torque and crowd force to rotate and push a special displacement tool into the ground to create the hole into which the pile will be placed. The tool, as shown in Fig. 1, consists of segments of different shapes, each of which is performing a certain function, as briefly described in this figure. As the tool advance into the ground, the loosened soil is pushed outwards thereby compacting the surrounding soil and consequently, the soil is stiffened and the shaft resistance of the pile enhanced.



Re-densification on retrieval
Counter-rotating flight to density any loose material in the hole during the tool retrieval

Stabilization
Cylindrical part for stabilization of the displaced material

Densification
Cone shaped tem exerts a lateral pressure to densify the soil conveyed upward by the flight

Loosening
the soil is loosened by the starter auger and conveyed upwards by the flights

Fig. 1 – FDP displacement tool.

Once the design depth is reached, the tool is withdrawn by counter rotation, while concurrently placing concrete and further densifying the soil. How quick or slow the drilling tool moves into the ground depends on the softness of the encountered soil. However, due to the complex nature of the operation, it is quite difficult to relate the soil behavior to drilling parameters such as torque, and rate of advancement. The use of advanced nonlinear regression tools, such as neural networks, could potentially be useful in overcoming this difficulty.

3. Neural Network Modeling

Neural networks are considerably simplified models of a biological brain capable of exploring complex relationships between variables. The basic structure of neural networks consists of simple processing units (neurons) connected by weighted links (synapses) which govern the information flow from the input to the output ends. The parallel nature of their architecture makes them quite capable of handling complex modeling tasks notwithstanding the simple nature of their processing elements. Neural networks began to gain popularity as alternative modeling tools in the aftermath of the introduction of back-propagation approach to neural network training by Rumelhart *et al.* (1986). Since then, a great deal of research work has been carried out on improving their performance and their application to a wide range of disciplines. Neural networks have been successfully applied to a variety of civil engineering problems such as geo-material modeling (Ellis *et al.*, 1995; Penumadu & Jean-Lou, 1997), soil–structure interaction (Shahin *et al.*, 2002; Hashash *et al.*, 2003) and erosion problem (Bateni *et al.*, 2007b).

In the present work, self –evolving neural network is used to predict the drilling torque required to displace the soil during the installation of a full displacement pile.

3.1. Self-Evolving Networks

The most widely used neural network is the back-propagation neural network (BPN), which uses sigmoid transfer function as processing unit and back-propagation (BP) algorithm in prediction error correction. Other types of neural networks are Random Boolean Networks (RBN), polynomial networks and high-order networks. The key challenge associated with neural network design is how to arrive at an optimimum network structure and parameters. the complex nature of the topology space makes it extremely challenging to optimize network architecture and arrive at the best simultaneously combination of network parameters (Miller *et al.*, 1989). The classical topology optimization techniques, such as Network Pruning and Incremental Learning Algorithm, tend to get the network entrapped in the topology space local minima (Angeline *et al.*, 1994). In this research, a population based self-evolution algorithm is used. The process of developing such a network begins by initially training a network with single hidden node, then gradually evolving in size as the learning process proceeds. The self evolution process begins by generating a population of neural nets with each having a random set of connection and synaptic parameters. The connection parameters are binary, assuming a value of 1 if there is a connection between two nodes and 0 if otherwise (Fig. 2). The binary parameters for each network in the population are updated using a jumping particle swarm optimization (JPSO) procedure. JPSO algorithm, developed by Martınez-Garcıa and Moreno-Perez (2008), is a discrete optimization algorithm which tends to perform better in finding the best solution in discrete space compared to discrete particle swarm optimization algorithm (DPSO) proposed by Kennedy and Eberhart (1997). In JPSO algorithm, the particle jumps from its current position to a new position under the influence of particle's experience, global best position and explorative tendency.. The particle's position is stochastically updated as follows

$$\boldsymbol{x}_{t+1} = \lambda_1 \otimes \boldsymbol{x}_t \oplus \lambda_2 \otimes \boldsymbol{b} \oplus \lambda_3 \otimes \boldsymbol{g} \tag{1}$$

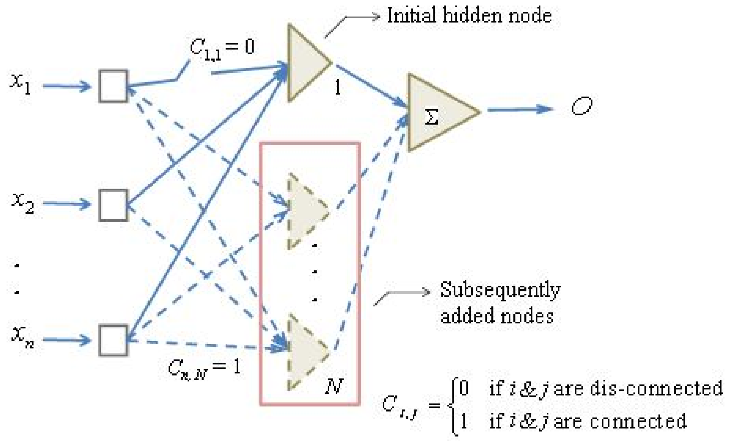

Initial hidden node
$C_{1,1} = 0$
x_1
x_2
x_n
1
Σ
O
Subsequently added nodes
$C_{n,N} = 1$
N
$C_{i,j} = \begin{cases} 0 & \text{if } i \,\&\, j \text{ are dis-connected} \\ 1 & \text{if } i \,\&\, j \text{ are connected} \end{cases}$

Fig. 2 – Topology of self-evolving network.

where $\boldsymbol{x}_t$ and $\boldsymbol{x}_{t+1}$ are the vectors of current and future particle positions in the discrete search space. The parameters λ_1, λ_2 and λ_3 are probabilities of jumping randomly, towards the best particle position and to the best swarm position respectively. ***b*** and ***g*** are, respectively, the particle best and global best positions. The updated position could be worse than the current one, therefore a random local search around the updated position is carried out to find a better solution. The local search is conducted using few steps of back-propagation algorithm due to the mixed nature of simultaneous optimization of network topology, activation functions and network parameters. The proposed JPSO algorithm is represented by the flowchart in Fig. 3.

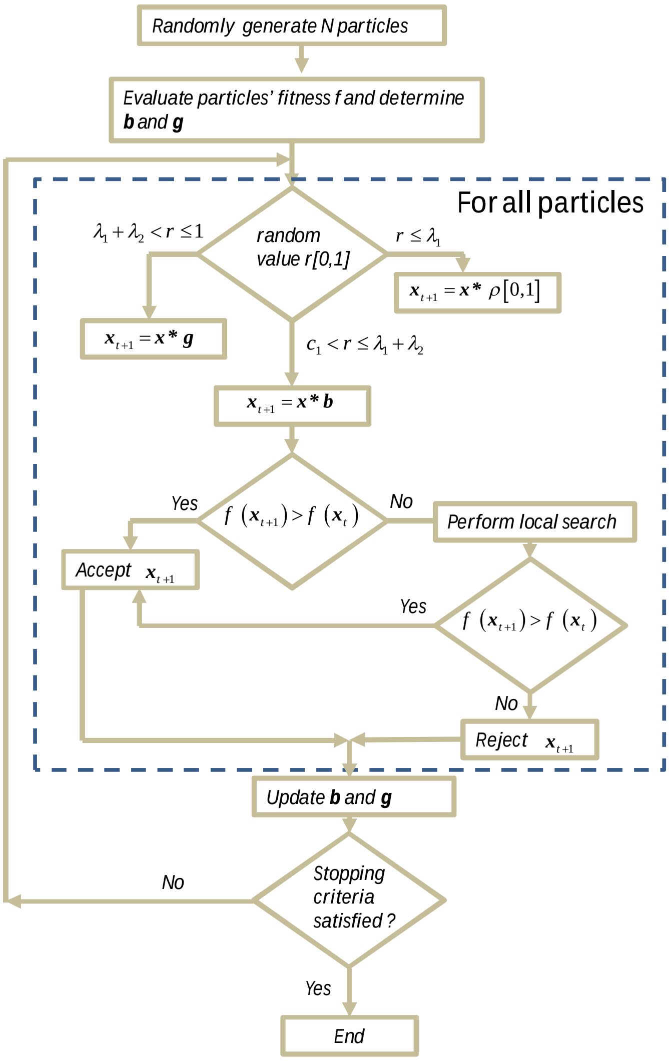

Randomly generate N particles
Evaluate particles' fitness f and determine **b** and **g**
For all particles
$\lambda_1 + \lambda_2 < r \le 1$
random value r[0,1]
$r \le \lambda_1$
$\boldsymbol{x}_{t+1} = \boldsymbol{x} * \rho[0,1]$
$\boldsymbol{x}_{t+1} = \boldsymbol{x} * \boldsymbol{g}$
$c_1 < r \le \lambda_1 + \lambda_2$
$\boldsymbol{x}_{t+1} = \boldsymbol{x} * \boldsymbol{b}$
Yes
$f(\boldsymbol{x}_{t+1}) > f(\boldsymbol{x}_t)$
No
Perform local search
Accept $\boldsymbol{x}_{t+1}$
Yes
$f(\boldsymbol{x}_{t+1}) > f(\boldsymbol{x}_t)$
No
Reject $\boldsymbol{x}_{t+1}$
Update **b** and **g**
No
Stopping criteria satisfied ?
Yes
End

Fig. 3 – Flowchart describing JPSO algorithm.

The synaptic weights of individual networks in the population are updated using a combination of particle swarm optimization (PSO) and BP algorithm. For the PSO part, the network parameters are updated using the eqs. proposed by Clerc and Kennedy (2002)

$$\boldsymbol{v}_{i,t+1} = \chi[\boldsymbol{v}_{i,t} + c_1 r_1 (\boldsymbol{b}_i - \boldsymbol{x}_{i,t}) + c_2 r_2 (g \ \ - \boldsymbol{x}_{i,t})], \tag{2}$$

$$\boldsymbol{x}_{i,t+1} = \boldsymbol{x}_{i,t} + \boldsymbol{v}_{i,t+1} \tag{3}$$

where: $\boldsymbol{v}_{i,t+1}$ is the updated velocity vector of particle *I*; $\boldsymbol{v}_{i,t}$ – the current velocity vector of particle *i*; $\boldsymbol{w}_{i,t+1}$ – the updated co-ordinates of particle *i*; $\boldsymbol{w}_{i,t}$ – the current co-ordinates of particle *i*; c_1 , c_2 – acceleration constants; r_1 , r_2 – uniformly distributed random numbers from 0 to 1; $\boldsymbol{b}_i$, g – the best particle experience and best swarm experience, respectively.

χ is defined as

$$\chi = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|}, \tag{4}$$

where $\varphi = c_1 + c_2 > 4$. The advantage of combining the two techniques is to benefit from the global search capability of the PSO and the ability of the BP algorithm to efficiently perform a local search. The algorithm involves initially training the network parameters using PSO for a certain number of iterations, then training some selected (best performing) particles among the swarm population using BP algorithm for few number of iterations. The results of the local search by BP algorithm are then used to update the positions of relevant particles and the PSO takes over again. The cycle is repeated until a sufficiently accurate is obtained.

When no further improvement is observed, the complexity of the network is increased by adding more nodes, one node at a time. To prevent the destruction of the so far acquired knowledge, the previous best particle positions (both topology and synaptic weights) are retained while adding one more node to the members of the swarm population. In this way, the computational burden of dealing with unnecessarily large networks is avoided as in the case of the algorithms proposed by Kiranyaz *et al.* (2009) and Xian-Lun *et al.* (2007), while at the same time avoiding the risk of getting stuck in the local minima of topology space.

3.2. Activation Function

The choice of suitable activation function plays a central role in successful development of neural networks. Sigmoid function has been the most widely used model for Artificial Neuronal Network (ANN) development due to its stability. However, despite the popularity it enjoys, it is not the most suitable under all conditions (Sopena *et al.*, 1999; Wong *et al.*, 2002). In this research, a combination of several functions is used as a processing function. The used processing function is expressed in the following eq.

$$f(x) = \sum_{i=1}^{i=n} k_i^{'} c_i \phi_i(x), \tag{5}$$

where: n is the number of sub-functions φ; c_i – an adaptive coefficient, while $k_i^{'}$ – a binary number; $\boldsymbol{x}$ – the vector of inputs to the node. The binary number assume a value of 1 if the associated functions are switched on and 0 if the functions are excluded. The adaptive coefficients are updated and trained alongside the synaptic weights, while the binary parameters are updated together with connection parameters using JPSO algorithm. In this manner, the topology, the synaptic weights and the activation functions are simultaneously optimized.

The sub-functions considered in this paper are as follows:

a) linear

$$\varphi_1(\boldsymbol{x}) = \left(\boldsymbol{k}^T \boldsymbol{w}\right)^T \boldsymbol{x} + b; \tag{6a}$$

b) wavelet

$$\varphi_2(\boldsymbol{x}) = \left[\left(\boldsymbol{k}^T \boldsymbol{w}\right)^T \boldsymbol{x} + a\right] \mathrm{e}^{-\left[\left(\boldsymbol{k}^T \boldsymbol{w}\right)^T \boldsymbol{x} + b\right]^2 / 2}; \tag{6b}$$

c) sinusoid

$$\varphi_3(\boldsymbol{x}) = \sin\left(\boldsymbol{w}^T \boldsymbol{x} + b\right); \tag{6c}$$

d) sigmoid

$$\varphi_4(\boldsymbol{x}) = \frac{1}{1 + \mathrm{e}^{\left(\boldsymbol{k}^T \boldsymbol{w}\right)^T \boldsymbol{x} + b}}, \tag{6d}$$

$\boldsymbol{w}$ represents the vector of synaptic weights of input signals, whereas a and b are the biases.

3. Neural Network Modeling

3.1. The Drilling Data

The database used in the present work consists of a total of 475 data sets based on the measurements taken during a full displacement piling operation. The measured parameters include the torque, rotational speed, rate of drilling progress and the crowd force. Also included in the database is the record of cone resistance tests carried out to determine the subsurface characteristics. The database cover a range of soil resistance values, and drilling parameters soil properties as summarized in Table 1.

Table 1
The Database Summary

Parameter	Set	μ	σ	$x_{\max}$	$x_{\min}$
T_q, [kN.m]	Training	44.997	15.063	94.900	23.500
	Testing	45.293	18.218	97.900	23.500
W_c , [kN]	Training	6.124	4.882	25.500	0.100
	Testing	5.985	5.185	25.500	0.100
v, [m/min]	Training	4.194	0.709	6.900	0.700
	Testing	4.082	0.911	5.800	0.700
ω, [rev/min]	Training	6.926	33.662	452.300	1.200
	Testing	6.107	17.331	122.500	1.200
q_c, [MN.m^{-2}]	Training	5.479	5.620	19.962	1.000
	Testing	5.101	5.447	19.917	1.014
r_f, [%]	Training	3.715	2.759	24.027	0.093
	Testing	3.714	2.685	21.319	0.384

3.2. Input Parameters

A reasonable neural network prediction can only be made if the input variables to the network are adequately representative of the factors governing the system under investigation. This is due to total dependence of the ANN on the observed behavior. Considering the full displacement piling, the drilling torque required to advance the bit into the subsurface is related to the drilling parameters such as the crowd force, W_c , rotational speed of the drilling tool, ω, and rate of tool advancement, v. Similar parameters were used by Gui *et al.* (2002), in their attempt to develop a correlation between soil characteristics and drilling variables. The soil resistance, represented here by cone resistance, q_c , and friction ratio, r_f, are also used as inputs to the model in order to provide an information about the soil deposit into which the drilling hole is constructed.

The torque, expressed as a function of the aforementioned parameters, can be represented by the following equation:

$$T_q = \Phi\left(W_c, \omega, v, q_c, r_f\right), \tag{7}$$

where Φ is an unknown function to be approximated using ANN. The five variables in the enclosed bracket, in eq. (7), serve as input parameters to the network, while the drilling torque is the network output.

3.3. Network Training and Validation

The database was partitioned into training and testing (validation) sets. A total of 347 data sets were used for training, while 128 sets were used for testing. To avoid a poor network generalization, the data is split in such a way that both the training and testing data belong, statistically, to the same population. The parameters used in assessing the prediction quality in the case of both training and testing include the root mean square error (*N*-RMSE), the mean (μ_λ) and standard deviation (σ_λ) of ratio of predicted torque to measured torque ($\lambda = T_{q\ \text{Pred.}}/T_{q\ \text{Measu}}$) are also used as prediction quality indicators. The mean value of λ ratio gives some information about whether or a model, on average, underestimates or overestimates the value in perspective. The standard deviation serves as an indicator of level of scatter in the prediction. A perfect model with 100% accuracy will have a mean value of 1.0 and a standard deviation of zero. The self-evolving network is trained and tested along with conventional fully connected BPN and GRNN networks. This is to provide a basis for assessing the relative performance of the former.

The optimized network can be represented by the following empirical relationship:

$$T_q\left(kNm\right) = 19.58\left(I_1 + I_2 + I_3\right), \tag{8}$$

where

$$I_1 = w_{13}x_3 + c_1 \sin\left(w_{22}x_2 + w_{42}x_4 + c_2\right) + c_3,$$

$$I_2 = \frac{c_4}{1 + \mathrm{e}^{-\left(w_{23}x_2 + w_{33}x_3 + w_{43}x_4 + c_5\right)}} + \frac{c_6}{1 + \mathrm{e}^{-\left(w_{54}x_5 + c_7\right)}},$$

$$I_3 = c_8\left(w_{15}x_1 + w_{25}x_2 + w_{45}x_4 + w_{55}x_5 + c_9\right)\mathrm{e}^{-\frac{\left(w_{15}x_1 + w_{25}x_2 + w_{45}x_4 + w_{55}x_5 + c_{10}\right)^2}{2}}.$$

The values of the network parameters are given in Table 2. All input values are scaled as follows:

$$x_{\text{scaled}} = \frac{x_{\text{actual}} - x_{\min}}{x_{\max} - x_{\min}}; \tag{9}$$

the maximum and minimum values can be looked up in Table 1.

Table 2

The Free Parameter Values of Self-Evolving Net Described in Eq. (8)

Input parameter	W_c	v	ω	q_c	r_f
Scaled parameter	x_1	x_2	x_3	x_4	x_5

Free parameter	Value	Free parameter	Value
w_{13}	–0.75929	c_1	–0.22947
w_{22}	–9.46171	c_2	12.64110
w_{42}	–3.40167	c_3	1.37908
w_{23}	–15.13870	c_4	–0.18727
w_{33}	–14.43390	c_5	5.16950
w_{43}	1.61375	c_6	–0.10398
w_{54}	13.17460	c_7	6.38598
w_{15}	–42.75290	c_8	–0.13878
w_{25}	–132.84600	c_9	–40.55960
w_{45}	1,012.37000	c_{10}	1.33719
w_{55}	403.19000		

3.4. Performance Assessment

The results of ANN predictions are plotted *vs*. the training and testing data in the scatter-grams shown in Figs. 4 *a* and 4 *b*, respectively. As can be seen from these figures, the agreement between the proposed network predictions and both training and testing data is quite reasonable as most of the data points seem to fall within 20% error level in both cases. The figures are indicative of the networks ability to learn from the training data and to accurately predict the output parameter (torque) in comparison with the test data. The low scatter level (σ_λ = 0.1299) confirms the good agreement between the model's prediction and the test data. The results of training and testing of several network types are compared with the result of self-evolving net in Table 3. It can be seen that all other networks return lower error value than the optimized network. However, the testing results indicate that the better agreement between the training data and the predictions of BPN and GRNN

Table 3

Training and Testing Results for Various Types of Network

Network type	No. of nodes	No. of parameters	*N*-RMSE (training)	*N*-RMSE (testing)
Self-evolving Network	3	20	0.0772	0.0878
BPN (sinusoid activation)	5	35	0.0754	0.1073
BPN (sigmoid activation)	5	35	0.0686	0.2037
BPN (wavelet activation)	5	40	0.0707	0.0917
GRNN	343	353	0.0448	0.0991

network amounted to over-learning as their predictions in comparison with testing data is inferior to the optimized network. The chart in Fig. 5 compares

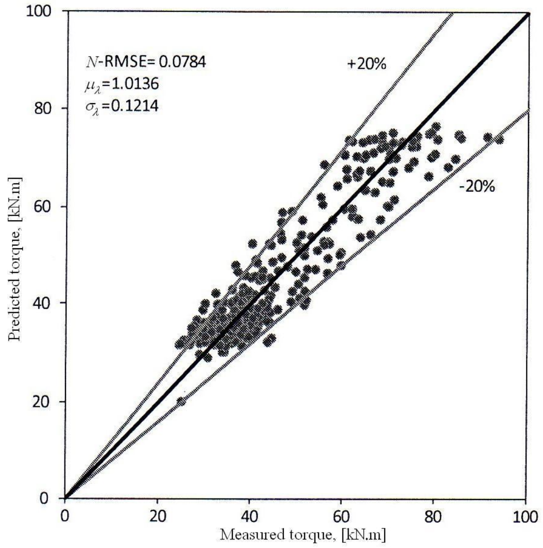


Fig. 4 *a* – Comparison of measured torque (training data) and self-evolving model predictions.

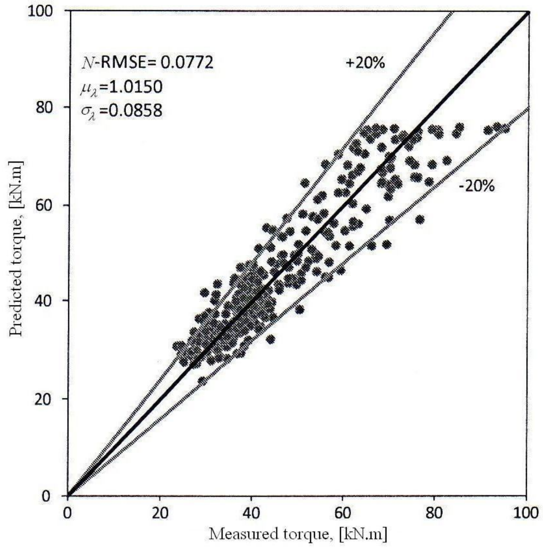


Fig. 4 *b* – Comparison of measured torque (testing data) and self-evolving model predictions.

the coefficients of the networks determination considered in this study with respect to test data. The chart indicated that the self-evolving network model

(SEANN) returns the highest value of R^2 (0.8834) in harmony with the results summary in Table 3.

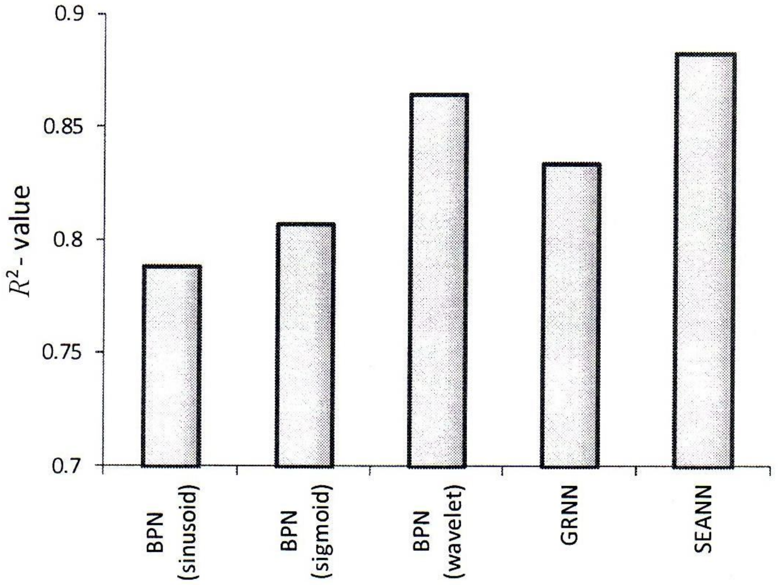


Fig. 5 – Performance comparison of various network models based on test data.

The accuracy alone is not sufficient to judge the performance of neural networks. Network size (in terms of number of free parameters) counts a lot in assessing the model efficiency. The self-evolving network returns the smallest number of parameters based on Table 3 results. With number of free parameters a little more than half of the number of parameters required in the case of sigmoid and sinusoid BPNs and exactly half of wavelet BPN parameters, the self-evolving network manages to make the most accurate result. The GRNN is on the most inefficient in terms of number of parameters. With such a staggering number of parameters (353), it still achieves a much lesser accuracy than both wavelet BPN and optimized network.

To examine the influence of various parameters involved in the modeling the torque, a sensitivity analysis is conducted by removing a parameter from the input set and evaluating the model performance without the parameter taken away. The proceeding is repeated until all parameters are considered The analysis results are shown in Fig. 6, which indicates that all parameters considered as inputs are found to affect, with a varying degree of influence, the magnitude of torque. It can also be seen from the figure that the rate of tool advancement, *v*, is the most dominant parameter influencing the output, followed by friction ratio, r_f, and cone resistance, q_c. The model turns out to be least sensitive to the rotational speed, ω, and the crowd force, W_c.

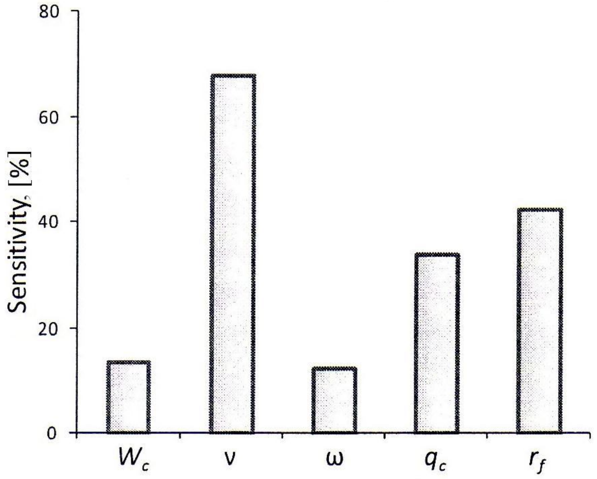


Fig. 6 – Model sensitivity to various input parameters.

4. Conclusions

The ability to predict the necessary torque for the completion of a given pilling work can prove to be a very useful tool in the hands of a specialist contractor. Since the main cost of the pilling works is due to fuel and onsite maintenance and repair it becomes obvious that a reasonable estimation of energy consumption in the course of the project will significantly help in ensuring cost control during the job execution. In full displacement piling operation, torque is the most important parameter in assessing the energy requirement. This paper proposes neural network model to estimate the amount of torque required to drive the full displacement tool into a silty deposit. The informations required by the empirical model include the crowd force, the rate of tool advancement and the tool's rotational speed. Other parameters include cone resistance and friction ratio of the soil subject to drilling. the model was developed based on the BP/PSO hybrid parameter optimization algorithm and JPSO based topology optimization technique. Based on its performance, the proposed model was found to agree well with an independent experimental data which was not included in the training process. The input parameters used in this study seemed significantly affect the estimated torque, based on the sensitivity analysis carried out in this study. Also, comparisons with the various BPN networks as well as GRNN model showed that the self-evolving model's performance, based on its accuracy and number of free parameters, is the best.

Acknowledgements. This paper was realized with the support of Posdru Cuantumdoc "Doctoral Studies for European Performance in Research and Innovation" ID79407 project funded by the European Social Fund and Romanian Government.

REFERENCES

Angeline P. J., Saunders G.M., Pollack J.B., *An Evolutionary Algorithm That Constructs Recurrent Neural Networks*. IEEE Trans. on Neural Networks, **5**, 54-65 (1994).

Basu P., Prezzi M., *Design and Applications of Drilled Displacement (Screw) Piles*. Publ. FHWA/IN/JTRP-2009/28, Joint Transp. Res. Program, Indiana Dept. of Transp. a. Purdue Univ., West Lafayette, Indiana, USA ,2009.

Bateni S.M., Jeng D-S., Melville B.W., *Bayesian Neural Networks for Prediction of Equilibrium and Time-Dependent Scour Depth Around Bridge Piers.* Adv. Eng. Software, **38**, 102–11 (2007b).

Clerc M., Kennedy J., *The Particle Swarm-Explosion, Stability and Convergence in a Multidimensional Complex Space*. IEEE Trans. on Evol. Comp., **6**, *1*, 58-73 (2002).

Ellis K.G.W., Yao C., Zhao R., Penumadu D., *Stress-Strain Modelling of Sands Using Artificial Neural Networks.* J. Geotech. Engrg., ASCE, **121**, *5*, 429-435 (1995).

Gui M.W., Soga K., Bolton M.D., Hamelin J.P., *Instrumented Borehole Drilling for Subsurface Investigation.* J. of Geotechn. a. Geo-environ. Engng., (ASCE), **128**, *4*, 283-291 (2002).

Hashash Y. *et al*., *Systematic Update of a Deep Excavation Model Using Field Performance Data*. Comp. a. Geotechn., **30**, 477-488 (2003).

Kennedy J., Eberhart R.C., *A Discrete Binary Version of the Particle Swarm Algorithm.* Proc. of IEEE Conf. on Syst., Man, and Cybernetics, Iscataway, New Jersey,USA, 4104–4109, 1997.

Kiranyaz S., Ince T., Yildirim A., Gabbouj M., *Evolutionary Artificial Neural Networks by Multi-dimensional Particle Swarm Optimization.* Neural Networks (in press).

Matınez Garcia F.J., Moreno Perez J.A., *Jumping Frogs Optimization: a New Swarm Method for Discrete Optimization*. Techn. Rep. DEIOC 3/2008, Dept. of Statistics, O.R. a. Comp., Univ. of La Laguna, Tenerife, Spain, 2008.

Miller G.F., Todd P.M., Hegde S.U., *Designing Neural Networks Using Genetic Algorithms.* Proc. 3rd Internat. Conf. Genetic Algorithms a. Their Applications, J. D. Schaffer (Ed.), San Mateo, CA, Morgan Kaufmann, 379–384, 1989.

Penumadu D., Jean-Lou C., *Geomaterial Modeling Using Artificial Neural Networks.* Artif. Neur. Networks for Civil Engin.: Fund. a. Appl., ASCE, 160-184 (1997).

Rumelhart D.E., Hinton G.E., Williams R.J., *Learning Internal Representation by Error Propagation.* Parallel Distributed Processing, Vol. **1**, Chap. 8, D.E. Rumelhart and J.L. McClelland (Eds.), MIT Press, Cambridge, Mass., USA, 1986.

Shahin M.A., Maier H.R., Jaksa M.B., *Predicting Settlement of Shallow Foundations Using Neural Networks*. J. of Geotechn. & Geoenvironm. Engng., ASCE, **128**, *9*, 785-793 (2002).

Sopena J.M., Romero E., Alquezar R., *Neural Networks with Periodic and Monotonic Activation Functions: A Comparative Study in Classification Problems.* Ninth Internat. Conf. on Artif. Neural Networks (ICANN '99), Edinburgh, UK, **1**, 1999, 323-328.

Wong K.-W., Leung C.S., Chang S.-J., *Use of Periodic and Monotonic Activation Functions in Multilayer Feedforward Neural NetworksTtrained by Extended Kalman Filter Algorithm.* IEEE Proc. Image Signal Proc., **149**, *4*, 217-224 (2002).

Xian-Lun T., Yon-Guo L., Ling Z., *A Hybrid Particle Swarm Algorithm for the Structure and Parameter Optimization of Feedforward Neural Networks.* Lecture Notes in Comp. Sci., 4493:213-218 (2007).

EVALUAREA MOMENTULUI MOTRIC NECESAR PENTRU REALIZAREA PILOȚILOR DE ÎNDESARE ÎN TERENURI SLABE FOLOSIND REȚELE NEURONALE

(Rezumat)

Se urmăreşte stabilirea unei rețele neuronale pentru estimarea momentului motric necesar la pătrunderea sculei de foraj pentru realizarea piloților de îndesare completă (FDP), pornind de la valorile rezistențelor obținute la penetrările cu con realizate în amplasament. Un algoritm rețea cu dezvoltare independentă este folosit la implementarea rețelei neuronale, pentru a reduce complexitatea modelului fără însă a prejudicia acuratețea acestuia. Baza de date folosită la realizarea modelului este constituită dintr-un set de înregistrări ale utilajului de realizat piloți FDP. Piloții au fost folosiți drept soluție pentru îmbunătățirea terenuui de fundare sub un decantor al Stației de Epurare a Apelor Uzate, Brăila. Estimările obținute prin folosirea rețelei neuronale sunt în concordanță cu cele înregistrate *in situ.*