Soil water content interpolation by hybrid harmony search - support vector machines model

MILAN ČISTÝ & MARTIN SUCHÁR

Water Management Department, Faculty of Civil Engineering STU in Bratislava, Radlinského 11, Slovakia

milan.cisty@stuba.sk, martin.suchar@stuba.sk

Abstract Hydrological processes are generally nonlinear and are governed by general physical laws. It is difficult to observe, describe and model them. Modeling, simulating and predicting water content in soil is essential for development of the agricultural information systems or in hydrology. One of the possibilities of management of soil moisture regime of agricultural land is application of the mathematical models. Quality of output from such models depends on the model inputs, especially climatic, meteorological, phenological, soil or topography data. Because the water content in soil is affected by many factors in its forecasting is difficult to achieve accurate results. Due to these problems are still necessary new and better methods to specify the resulting simulation. The authors describe in this contribution creation of the model utilizing support vector machine (SVM), which belongs to data driven classification and regression tools. For present work is characteristic application of the simulation model to interpolation of the measured values, which are available in 2-week intervals, to the daily values. Accuracy of the SVM application to predicting soil moisture in the five soil depth horizons is compared with the model created by artificial neural networks. In this testing SVM model showed higher accuracy and better handling.

Keywords soil moisture; forecasting; support vector machine; artificial neural network

INTRODUCTION

Soil water plays a key role in the transfer of energy and mass between land surfaces and the atmosphere, rivers, and aquifers. The spatial and temporal distribution of soil water is a critical part of many disciplines including agriculture, forest ecology, hydro-climatology, civil engineering, water resources, and an ecosystem modeling. Hydrological processes involved in creating the soil water regime include in particular precipitation, inflow and outflow of surface water inflow and outflow of groundwater, surface water infiltration into the soil, water leakage through the soil profile, moisture infiltration from groundwater, and others. Soil moisture regime, especially the active layer of soil is important from different aspects, e.g. decrease in soil moisture below a certain value, the soil water becomes less available to plants, etc. Long-term monitoring of soil water regime is necessary, from various aspects: protection against floods, the region's ecology, food production, production of biomass for alternative energy sources etc. This kind of work is time, equipment, staff and finances consuming. Monitoring of the soil moisture is necessary for described purposes, but in many cases can be complemented by other methods, including mathematical modeling. Analyzing such a large quantity of data involved in the soil water regime creation is a challenging task, so the demand for accurate and efficient algorithms are undeniably justified. Accuracy of modeling doesn't depend only on the input data but also on the selection of a suitable model. If a physical description of phenomenon is known and reliable data for it are available, standard models consist of systems of differential equations or other mathematical arsenal necessary to quantify the ongoing physical happening is used. However, it can happen that our knowledge of the process is limited or data are not representative enough. Sometimes the process is so complex that on present state of knowledge cannot be correctly mathematically described at all.

Artificial neural networks and other data-driven models such as SVM, can under certain conditions to enter into such gaps of mathematical description and replace it with the knowledge stored in the data. Their usage is based on the principle that from the known inputs and outputs (e.g. measured) they learn how to generate from input correct output. Then in application phase unknown outputs could be generated from known inputs.

The submitted work compares soil water content models based on neural networks (ANN) and support vector machines (SVM) methodology. Particular interest of paper is given upon alternative data-driven method SVM, which were first used as a classifier and then its functionality has been extended to regression. The advantage of ANN and SVM data-driven models is their ability to learn from the model data and the ability to generalize knowledge from them, often without detailed knowledge of various state variables of the process. That makes them suitable alternative tool to address complex processes. Moreover, the SVM formulate a quadratic optimization problem that avoids local minima problems, which makes them often superior to traditional (iterative) learning algorithms such as multi-layer perceptron (MLP) type of neural network.

METHODOLOGY

Support vector machine regression

Vapnik and his co-workers developed SVM in the early 1990s for classification applications. Vapnik (1995) later extended his work by developing SVM for regression. SVM is one of the so-called kernel-based methods of machine learning, having a sophisticated mathematical theory, which stands in their background (structural risk minimization principle etc.). SVMs are popular for their greater ability to generalize, which is the goal in statistical learning. The basic idea behind SVMs is to map the input space into a high dimensional feature space utilizing kernels. SVMs generally result in a function estimation equation of the form:

$$f(x,w)=\sum_{i=1}^{m} w_i \times \phi_i(x) + w_0 \tag{1}$$

where the functions $\{\phi_i,(\mathrm{x})\}^{\mathrm{m}}$ are feature space representations of the input query *x*; *m* is the number of patterns that contain all the information necessary to solve a given learning task, hereinafter referred to as support vectors; and w = {w_0 w_1, ... w_m} are the SVM parameters. The mapping of *x* by ϕ(x) into a high dimensional feature space is chosen in advance by selecting a suitable kernel function. The learning algorithm seeks to define a hyperplane that is necessary for applying the linear regression in the SVM formulation. Now the problem is to determine *w* and the corresponding *m* support vectors from the training data. To avoid the use of empirical risk minimization (e.g., quadratic residual function), which may result in overfitting, Vapnik proposed a structural risk minimization (SRM) in which one minimizes some empirical risk measure regularized by a capacity term. This is the most appealing advantage of SVMs. Consistent with SRM, therefore, the objective function of SVM formulation is to minimize the following:

$$E(w)= \frac{1}{M}\sum_{i=1}^{M}\left|y_i - f(x_i, w)\right|_{\varepsilon} + \frac{1}{2}\|w\|^2 \tag{2}$$

Vapnik employed the ε-insensitive loss function, $|y_i - f(x_i, w)|_{\varepsilon}$, wherein those differences between estimated output, $f(x_i, w)$, and the observed output, y_i , which lie within the range of $\pm \varepsilon$ do not contribute to the output error. Using the ε-insensitive loss function has shown that equation (2) is equivalent to the following dual form:

$$y = f(x,\alpha^*,\alpha) = \sum_{i=1}^{M}(\alpha_i^* - \alpha_i)K(x_i, x) + \lambda_0 \tag{3}$$

where the Lagrange multipliers α_i and α_i^* are required to be greater than zero for $i = 1, ..., M$, and $K(x_i,x)$ is a kernel function defined as an inner product in the feature space, $K(x_i, x) = \sum_{i=1}^{M}\phi(x_i)\cdot\phi(x)$.

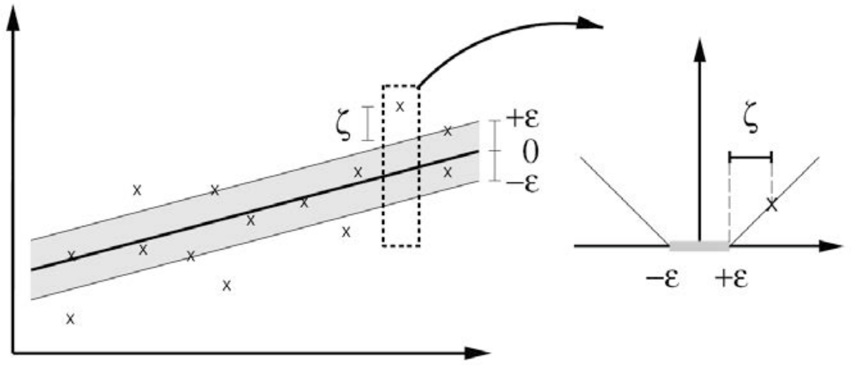

Fig. 1 Margin loss, which corresponds to linear ε-ignoring loss making function (Smola, Scholkopf, 1998)

SVM involves of a quadratic programming problem that can be solved efficiently and for which a global extreme is guaranteed. In working with statistical learning tools our ultimate goal is to estimate a functional dependency, f(x), between inputs $\{x_1,x_2,\ldots,x_l\}$, taken from $x \in R^K$, and $\{y_1,y_2,\ldots,y_l\}$ with $y \in R$ taken from a set L of independent and identically distributed (i.i.d.) observations. Hence, f(x) is estimated by minimizing the following regularized functional:

$$\text{Minimize } \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{L}(\zeta_i + \zeta_i^*) \tag{4}$$

$$\text{Subject to: } y_i - \sum_{j=1}^{K}\sum_{i=1}^{L} w_j x_{ji} - b \le \varepsilon + \zeta_i \text{ ; } \sum_{j=1}^{K}\sum_{i=1}^{L} w_j x_{ji} - b - y_i \le \varepsilon + \zeta_i^* \tag{5}$$

$$\text{Where } \sum_{j=1}^{K} w_j x_j + b \quad \text{with } w \in R^n, b \in R \tag{6}$$

where, K is the number of support vectors and ‘‘b’’ is the bias. The ‘‘w’’ term represents the weights for input (only the support vectors) to the output. The quantity ‘‘C’’ determines the trade-off between the complexity of the function f and the tolerance for error in the prediction of the function. Deviations of the machine's prediction from observed system behavior that are smaller than ε are allowed without penalty (we do not care about errors that are less than ε). This is known as the ε-insensitive loss function (also shown in Fig. 1). Also shown here are the slack variables, ζ_i and ζ_i*, that determine the degree to which samples will be penalized with errors larger than ε.

Equation (5) is solved in its dual form by employing Lagrange multipliers. The problem results in maximizing the following functional:

$$W(\alpha^*,\alpha) = -\varepsilon \sum_{j=1}^{L} (\alpha_i + \alpha_i^*) + \sum_{i=1}^{L} y_i (\alpha_i - \alpha_i^*)$$

$$-\frac{1}{2} \sum_{i,j=1}^{L} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) k(x_i - x_j) \qquad (7)$$

Subject to constraints

$$\sum_{i=1}^{L} (\alpha_i^* - \alpha_i) = 0, \quad 0 \le \alpha_i, \alpha_i^* \le C \qquad (8)$$

where, i = 1,….,L is the sample size, and the approximating function is

$$f(x) = \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) k(x, x_i) + b \qquad (9)$$

where, $\alpha*$ and α are Lagrange multipliers, and $k(x,x_i)$ is the kernel function. The choice of a kernel in SVM is analogous to the problem of choosing a suitable architecture in NN. Further details can be found in Vapnik (1995).

Multilayer perceptron

Artificial neural networks (ANN) are inspired by biological processes in the human brain respectively in the nervous system and applied to various technical problems for which sufficient, representative data are available. Because of their basic principle - to extract knowledge from data, they are called data-driven models. Generally ANN is defined as a computing system that has ability to learn and retain information (and relationships between them) and allows their further use. The most commonly used ANN is a multilayer perceptron (MLP). It's a feed forward network with a controlled type of learning. The input signals pass through this type of network only in forward direction, from input layer to output layer. The basic element of MLP is neuron, which generally has more inputs and one output. Neurons in the network are linked to each other and these connections transform the signal coming from the previous neurons by connection's weights. The sum of these weighted signals is then transformed by activation function of the neuron (nonlinear), which affects the output to the next neuron. MLP uses three or more layers of neurons – input layer, one or more hidden layers and output layer, all with nonlinear activation function. Nonlinearity included in this flow of the input signal (activation function, hidden layer etc.) allows that network could learn complex nonlinear tasks.

Application of ANN model is divided into three separate parts. The first is called learning phase and is about training the model with training input data. The actual output of the network must be known for this type of ANN in the learning stage. There are many algorithms for training neural networks; most of them can be viewed as a straightforward application of optimization theory and statistical estimation. Learning of the MLP is accomplished by method of error back propagation. Error in this sense means difference between expected and actual output of MLP. The signal transmitted between neurons is changed depending on adjustable parameters called weights as was mentioned in previous paragraph. The goal of the learning process is mainly to define these weights. Finding the appropriate network parameters is repeated until the error between the desired and actual output from ANN is minimal.

Details of back propagation learning method are described in the general literature on this subject (e.g. Kvasnicka et al. 1997) therefore we don’t deal with this in more detail here.

In second – verification phase of ANN application is trained network verified with test data (actual output of the network must be known in this stage too), and if this is accomplished with satisfactory results, model is ready to use for real application (where output data are unknown).

APPLICATION AND RESULTS

Description of the study area and the input data

For testing purposes of the methods described in previous section, data were taken from probe installed in the village Báč on Žitný ostrov (Slovakia). In this and other nearby locations moisture profiles in the unsaturated soil zone and ground water levels are monitored. In some of these localities since 1999 is running continuous monitoring, but from probe Báč are samples taken at two weeks intervals. That is why interpolation of these measured values to daily interval will be accomplished by data driven models presented in this study. Soil profile at the site Báč has complicated layered structure. There is loam on the surface, and passes into sandy loam and in depth of about 90-100 cm comes to sand and later into gravel soil. Moisture content of the soil profile was monitored using neutron probe in depth distances always 10 cm from each other. The measurements were made with 2-weeks frequency, only in winter measurements were accomplished once a month. At each site, calibration curves were made at different seasons. They served for the refinement of the computational relationships recommended by the manufacturer of neutron probe.

Measured moisture content of soils from five horizons in the probe Báč (0, 20, 30, 40, 50 cm) was used for the testing purposes. These data were collected at mentioned 2-weeks intervals during period April 1999 - December 2009. A sum of 189 data vectors is available, but for the calculation purpose were excluded data from the winter months from November to February. Thus 117 data vectors were used. These data about soil moisture are in further calculations used as dependent variables that will be calculated on the basis of data taken from meteorological station Gabčíkovo. From Gabčíkovo station this type of data were available: the average daily temperature, relative humidity, wind speed, sunshine and daily precipitation but after the correlation analysis and other considerations were only the average daily temperature and daily precipitation totals used. In the proposed 5 models described below are these two variables used as inputs. They are taken from different time lag before the date on which the value of soil moisture is computed (e.g. T_{t-3} is temperature 3 days before day in which is prediction of the soil moisture computed). In every model there is every variable from more than one day taken. This structure of input data have to introduce into the calculation what is in the SVM method missing - time dynamics, since the SVM model itself is basically static. As the input are in some models used also a sums of precipitation amounts for the previous 20 days and the average temperature for the same interval. These two variables are intended to represent the meteorological and soil-humidity condition in the study area, which affects consequences of rainfall and temperatures in the immediately preceding days to the soil moisture value.

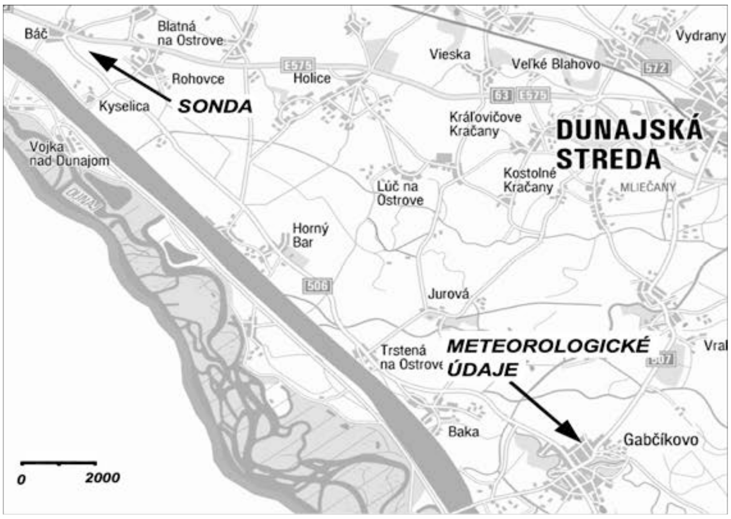

Fig. 2 Map of study area

The following data model structures were evaluated using SVM and MLP:

Model 1: $\theta_t = f(T_{t-1}, T_{t-2}, T_{t-3}, T_{t-4}, T_{t-5}, Z_{t-1}, Z_{t-2}, Z_{t-3}, Z_{t-4}, Z_{t-5}, U_{20}, T_{20})$

Model 2: $\theta_t = f(T_{t-1}, T_{t-2}, T_{t-3}, T_{t-4}, T_{t-5}, Z_{t-1}, Z_{t-2}, Z_{t-3}, Z_{t-4}, Z_{t-5})$

Model 3: $\theta_t = f(T_{t-1}, T_{t-2}, T_{t-3}, T_{t-4}, T_{t-5}, T_{t-6}, T_{t-7}, T_{t-8}, T_{t-9}, T_{t-10}, Z_{t-1}, Z_{t-2}, Z_{t-3}, Z_{t-4}, Z_{t-5}, Z_{t-6}, Z_{t-7}, Z_{t-8}, Z_{t-9}, Z_{t-10})$

Model 4: $\theta_t = f(T_{t-1}, T_{t-2}, T_{t-3}, Z_{t-1}, Z_{t-2}, Z_{t-3}, U_{20}, T_{20})$

Model 5: $\theta_t = f(Z_{t-1}, Z_{t-2}, Z_{t-3}, Z_{t-4}, Z_{t-5}, U_{20}, T_{20})$

To increase the accuracy of simulation a normalization of the input data was accomplished. Normalization should be made to eliminate that some variables will have greater impact on learning than others. In the process of normalization, all variables were transformed into range (- 1, 1) which guarantees that they will have equal importance in shaping the model. Normalization was carried by a linear relationship:

$$y = kx + q \tag{10}$$

where y is the normalized value, x is original value and:

$$k = \frac{2}{x_{\max} - x_{\min}} \qquad q = \frac{x_{\min} + x_{\max}}{x_{\min} - x_{\max}} \tag{11}$$

where x_{max} and x_{min} are the maximum and minimum values from one column of input data.

For the calculations has been used popular and frequently used libary Libsvm, from the authors Chih-Chung Chang and Chih-Jen Lin. Libsvm (A Library for Support

Vector Machines) is freely available software that can be obtained from the address http://www.csie.ntu.edu.tw/~cjlin/libsvm/. It is designed for classification (methods C-SVC, nu-SVC) or regression by support vectors (using ε-SVR, nu-SVR). As was already mentioned in method description the solution mainly lies in finding an appropriate regulatory parameter *C*, the tolerance band width *ε* and choosing an appropriate kernel function and its parameters. Based on recommendations from the literature and the experiments radial basis function was chosen:

$$k(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2), \mathit{pre}\ \gamma > 0 \tag{12}$$

where x_i and x_j are the input data and γ is the parameter of RBF function. RBF function parameter γ is necessary to set; in addition, it is necessary to determine the appropriate C and ε. The meaning of these parameters was explained in methodology section. On the basis of literature review it is possible to estimate the possible range for individual parameters but there isn’t a general rule for its accurate determination because this setting is problem depended. For optimal parameter setting harmony search heuristic methodology was used. A harmony search is a metaheuristic search algorithm inspired by the improvisational process of musicians introduced by Geem (Geem et al., 2002). In the HS algorithm, each musician corresponds to one decision variable; a musical instrument’s pitch range corresponds to a decision variable’s value range; the musical harmony at a certain time corresponds to a solution vector at certain iteration; and the audience’s aesthetics corresponds to an objective function. Just like a musical harmony is improved time after time, a solution vector is improved iteration by iteration by the application of the improvisation’s operators (the random selection of a tone, a musician’s memory considerations or a pitch adjustment). For each combination of parameters (γ, C, ε) generated in search process a model using this actual combination of parameters based on training data is created and as a criterion for selecting appropriate combinations of parameters the correlation coefficient is calculated but for the application of created model on the testing data. This is also value of the objective function of the harmony search methodology.

Setting the ANN was made by trial and error process with various settings (different numbers of neurons in the hidden layer, learning rule either momentum or Levenberg-Marquardt and other parameters) and by this way was found suitable architecture of ANN (with one hidden layer containing 5 neurons, activation function hyperbolic tangent in hidden layer and in output layer was linear function selected). As a learning rule was chosen Levenberg-Marquardt algorithm and the calculations were carried out by neural network simulator NeuroSolutions.

Results were statistically evaluated by mean square error (MSE) and correlation coefficient (R).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\theta_i - \theta_i^m)^2 \tag{13}$$

$$R = \frac{\sum_{i=1}^{n}(\theta_i - \bar{\theta})(\theta_i^m - \bar{\theta})}{(n-1)\sigma.\sigma_m} \tag{14}$$

where: θ is the measured value θ_m is the predicted value $\bar{\theta}$ is the average value, σ and σ_y are standard deviations of measured and modeled data, and n is the number of data information.

The results for five data models mentioned using MLP are evaluated in Table 1 and for SVM in Table 2.

Table 1 MLP model evaluation by R and MSE.

Model		10 cm	20 cm	30 cm	40 cm	50 cm	Sum
M1	R	0.88913	0.84224	0.80083	0.71828	0.71631	3.96679
	MSE	0.0012	0.00117	0.00164	0.00237	0.00184	0.00823
M2	R	0.67119	0.70037	0.71485	0.61335	0.59648	3.29625
	MSE	0.00314	0.0025	0.00224	0.00295	0.00237	0.0132
M3	R	0.71331	0.66752	0.60884	0.55622	0.57566	3.12154
	MSE	0.00321	0.0032	0.00376	0.00426	0.00264	0.01708
M4	R	0.87799	0.87044	0.84003	0.83933	0.85078	4.27857
	MSE	0.00123	0.00106	0.00131	0.00152	0.00121	0.00633
M5	R	0.66573	0.73933	0.75103	0.7762	0.78289	3.71518
	MSE	0.00683	0.00329	0.00221	0.0018	0.00127	0.0154

Table 2 SVM model evaluation by R and MSE.

Model		10 cm	20 cm	30 cm	40 cm	50 cm	Suma
M1	R	0.86771	0.88636	0.85367	0.83117	0.83071	4.26961
	MSE	0.00186	0.00101	0.00121	0.00377	0.00128	0.00912
M2	R	0.77936	0.79473	0.75957	0.73924	0.74213	3.81503
	MSE	0.00326	0.00154	0.0038	0.00305	0.00306	0.01471
M3	R	0.8159	0.84076	0.81171	0.80222	0.79241	4.06299
	MSE	0.00188	0.00127	0.00283	0.00181	0.00192	0.00971
M4	R	0.86495	0.87843	0.85572	0.8386	0.82778	4.26548
	MSE	0.00152	0.0011	0.0012	0.00133	0.00123	0.00637
M5	R	0.85857	0.88455	0.86213	0.8386	0.84248	4.28632
	MSE	0.00464	0.00303	0.00361	0.00133	0.00304	0.01566

Correlation coefficients for the calculations using MLP ranged from 0.55622 to 0.88913, with the average value 0.735. For SVM the correlation coefficients range from 0.73924 to 0.88636 with average value 0.828. For this reason, was from previously mentioned calculations selected for the final interpolation the SVM model as preferable and settings of input data into a final model such as in the first model (M1). In addition, the authors consider the SVM model as preferable because it does not suffer from the problem of accidental falling into local minima. On Fig. 3 are evaluated daily values of moisture at a depth of 20 cm for the growing season in 2007.

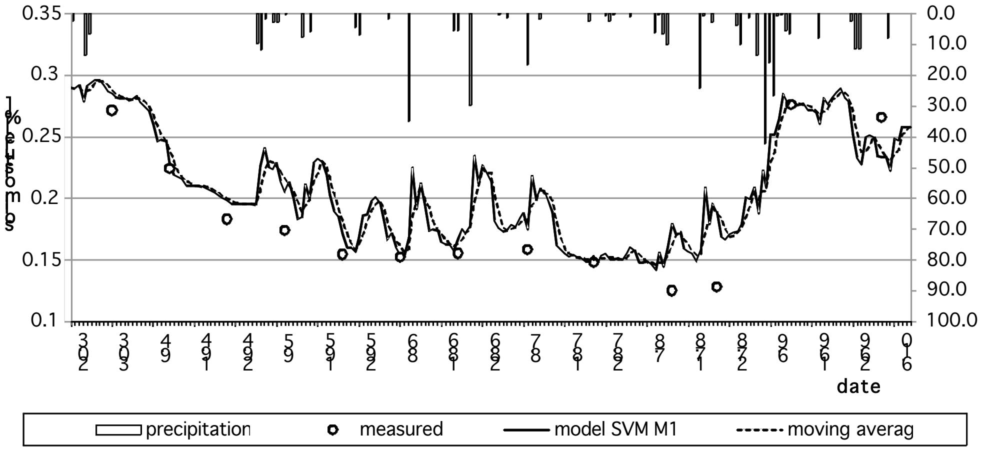

Fig. 3 Interpolation of the measurements of soil moisture to daily values, year 2007 for underground level 20 cm

CONCLUSION

Development of data management systems based on machine learning, make in the last decade significant progress, and these models will find their application in water management and hydrology, where are used for the prediction or regression tasks. In the presented work we compare the model for soil water content simulation and interpolation based on MLP methodology with newer types of data-driven model SVM. For both methods have been used same training and test input data set. The soil moisture data used to construct the models were measured in the probe Bač (Slovakia), and meteorological data from the measuring station Gabčíkovo. The results of both models are compared graphically and using two statistical coefficients (Tables 1, 2). From this comparison are the results of SVM method more accurate, which confirms their better potential for application in this task because their results are stable and repeated calculation do not change, unlike the MLP, where the results do not guarantee stability and change in the recalculation process. Both methods could be used as an alternative method to standard measurements and simulation.

Acknowledgments This work was supported by the Slovak Research and Development Agency under the contract No. LPP-0319-09, APVV-0443-07, APVV-0271-07 and by the Scientific Grant Agency of the Ministry of Education of the Slovak Republic and the Slovak Academy of Sciences, Grant No. 1/0585/08.

REFERENCES:

Cortes, C., Vapnik N. V. (1995) Support-vector networks. Machine Learning, 20 (3), 273–297.

Čistý, M., (2009) Comparison of the data-driven models for soil water regime analysis. In: People,buildings and environment 2009 : International scientific conference, ČR, 26.-27.11.2009, Brno, 55-62.

Geem, Z.W., Kim, J.H., Loganathan, G.V. (2002) Harmony search optimization: application to pipe network design, *International Journal of Modeling and Simulation,* vol. **22**(2), pp. 125-133.

Chang, C.C., Lin, C. J. (2001) Libsvm: a library for support vector machines.

Kvasnička, V., Beňušková, Ľ., Pospíchal, J., FarkaŠ, I., Tiňo, P., Kráľ, A. (1997) Úvod do teórie neurónových sietí, Iris, Bratislava, ISBN 80-88778-30-1.

Kohonen, T. (2001) Self-Organizing Maps. Springer-Verlag, Berlin.

Smola, J. A, Scholkopf, B. (1998) A tutorial on Support Vector Regresion, In: NeoroColt2 Technical Report Series, NC2-TR-27150.

Suchár, M., Čistý, M. (2009) Predpovedanie obsahu vody v pôde pomocou učiacich algoritmov, *Acta hydrologica slovaca,* Ročník **10**, č. 2, 2009, 266 – 274, EV 3224/09, ISSN 1335-6291.

Vapnik, N.V. (1995) The Nature of Statistical Learning Theory, In: Springer-Verlag New York, Inc.

Vesanto, J. (2002) Data Exploration Process Based on the Self-Organizing Map Espoo, Finland on the 16th of May, 2002, (ISBN 951-22-5897-8).

Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J. (1999) Self-Organizing Map in Matlab: the SOM Toolbox. In Proceedings of the Matlab DSP Conference 1999, Espoo, Finland, pp. 35-40. © 1999 Comsol Oy.

Vesanto, J., Alhoniemi, E. (2000) Clustering of the Self-Organizing Map. *Transactions on Neural Networks*, Vol. **11**, Number 3, pp. 586-600. © 2000 IEEE.