Extended Self-Organizing Maps with Local Linear Mappings for Function Approximation and System Identification

Dimitrios Moshou[(1)], Herman Ramon

Department of Agro-Engineering and Economics, Laboratory of Mechanical Engineering, K.U. Leuven, Kardinaal Mercierlaan 92, 3001 Heverlee, BELGIUM
[(1)] tel: +32-16-321478, fax: +32-16-321994, email: dimitrios.moshou@agr.kuleuven.ac.be

Abstract
A new method for function approximation and system identification based on the Self-Organizing Map is presented. The standard Self-Organizing Map (SOM) is extended with Local Linear Mappings to enable the original algorithm to learn input-output relationships with reasonable accuracy. To every node in the SOM along with the input weight two output weights are assigned: one that stores the output part of an input-output pair and one that stores the local gradient matrix (Jacobian) of the input-output mapping that is calculated from the training pairs. A training algorithm for the Jacobian matrices is derived based on a Least Squares approximation by utilising properties of the pseudoinverses of vectors. The method is tested in function approximation of a multivariable function, system identification of a highly nonlinear system and system identification of a hydraulic actuator based on input-output measurements.

I. Introduction

The Self-Organizing Map commonly referred to as SOM [Kohonen 1982] is a method that converts complex, nonlinear statistical relationships between high-dimensional data into simple geometric relationships. However the original SOM is known to perform poorly on regression problems because of the discrete representation of the data. Also it cannot by itself represent input-output relationships. By extending the SOM with output weights that store the output part of a mapping together with local gradient information, a first order expansion around the representative output can provide the original algorithm with the ability to perform function approximation and system identification. Such a network is generally called an LLM (Local Linear Map) network. LLM networks have been introduced earlier [Ritter 1991]; [Ritter et al., 1992]. Similar learning algorithms have already been applied in time-series prediction [Martinetz et al., 1993] and in non-linear prediction [Walter et al., 1990].

II. Extended SOM with LLMs

The Self-Organizing Map [Kohonen 1995] is a neural network that maps signals ($\mathbf{x}$) from a high-dimensional space to a one- or two-dimensional discrete lattice of neuron units ($\mathbf{s}$). Each neuron stores a weight ($\mathbf{w}_s$). The map preserves topological relationships between inputs in a way that neighboring inputs in the input space are mapped to neighboring neurons in the map space. When extended with output weights ($\mathbf{y}_s$) it can actually learn in a supervised way the mapping $\mathbf{y}=f(\mathbf{x})$. By utilising Local Linear Mappings, the approximation accuracy is increased because gradient information of the actual function is used to produce a first-order expansion around the representative output weight leading to the approximation

$$\mathbf{y}_{net}=\mathbf{y}_s+\mathbf{A}_s(\mathbf{x}-\mathbf{w}_s) \quad (1)$$

where with $\mathbf{A}_s$ denotes an approximation of the local gradient of function $f(\mathbf{x})$ around $\mathbf{w}_s$. The learning algorithm for the input and output weights is derived from the original Kohonen algorithm [Kohonen 1982] and has as:

$$\Delta \mathbf{w}_s^{(in)} = \varepsilon h(\mathbf{x}-\mathbf{w}_s^{(in)}) \quad (2)$$
$$\Delta \mathbf{w}_s^{(out)} = \varepsilon h'(\mathbf{x}-\mathbf{w}_s^{(out)}) \quad (3)$$

where ε, ε΄ and h, h΄ are the learning rates and the neighborhood kernels respectively. The neighborhood kernels used had the form of a Gaussian like:

$$h = \exp(-\equiv \mathbf{x}-\mathbf{w_s} \equiv^2 / \sigma^2) \quad (4)$$

where ≡.≡ denotes the Euclidean norm and σ denotes the variance of the Gaussian distribution. The learning algorithm for the Jacobian matrices ($\mathbf{A_s}$) is derived from Kohonen's learning rule. The updating of the output weights is performed also the enhanced approximation that uses the first-order expansion (1). This leads to the updating equation:

$$\Delta \mathbf{y}_{net} = \varepsilon'' h'' (\mathbf{y}-\mathbf{y}_{net}) \quad (5)$$

By assuming that $\mathbf{y_s}$ has already been updated, so for this purpose it is kept constant, the first order approximation (1) substituted in (5) gives:

$$\mathbf{As}^{(new)} (\mathbf{x}-\mathbf{ws}^{(in)}) = \mathbf{As}^{(old)} (\mathbf{x}-\mathbf{w_s}^{(in)}) + \varepsilon' h'' (\mathbf{y}-\mathbf{y_{net}}) \quad (6)$$

By using properties of the pseudoinverses of vectors the updating scheme for the Jacobian matrices becomes:

$$\Delta \mathbf{A_s} = \varepsilon' h'' (\mathbf{y}-\mathbf{y_{net}})(\mathbf{x}-\mathbf{w_s})^+ \quad (7)$$

where $^+$ denotes the pseudoinverse and ε΄΄, h΄΄ are the learning rate and the neighborhood kernel respectively. It must be noted that (7) is a least squares approximation of (6). Hence the learning algorithm for the Jacobian matrices becomes:

$$\Delta \mathbf{A_s} = \varepsilon' h'' (\mathbf{y}-\mathbf{y_{net}})(\mathbf{x}-\mathbf{w_s})^T / \equiv \mathbf{x}-\mathbf{w_s} \equiv^2 \quad (8)$$

In the following applications equations (2), (3) and (8) are used as the updating equations and the network estimate is calculated from (1).

III. Function Approximation

In this application, input-output data from a multivariable function are used to train an LLM SOM using equations (2), (3) and (7). The function that was used in this example is a simple two dimensional Gaussian of the form:

$$y = \exp(-\equiv \mathbf{x} \equiv^2 / \sigma^2) \quad (8)$$

where $\mathbf{x} \in \Re^2$ and σ denotes the variance of the Gaussian distribution. The inputs of this function were selected to lie in the set [-0.3,0.3]x[-0.3,0.3] and σ was chosen to be equal to 0.1. The estimate of the network at every time step was calculated from (1). Two training sessions, one for a 6x6 map and one for a 10x10 map, were performed. The 6x6 map was trained for 10000 epochs while the 10x10 map was trained for 20000 epochs. The training parameters ε,σ,ε΄,σ΄,ε΄΄, and σ΄΄ had the time dependence $x(t) = x_i(x_f/x_i)^{t/tmax}$ according to [Martinetz et al., 1990]. Initial and final values for the training parameters had as $\varepsilon_i=\varepsilon_i'=\varepsilon_i''=1$, $\varepsilon_f=\varepsilon_f'=\varepsilon_f''=1$ and $\sigma_i=\sigma_i'=\sigma_i''=0.4$x(number of nodes in one dimension of the map) and $\sigma_f=\sigma_f'=\sigma_f''=0.02$x(number of nodes in one dimension of the map). The input and output weights and the Jacobian matrices were initialized with small random values which were not important for the convergence of the map. The error measure used in this case is the average

Euclidean distance for 100 testing points taken from the same input distribution as that used for training. This evaluation was performed after every 100 training steps in order to monitor the convergence of the training algorithm. The final values for these error measures were equal to 0.009 for the 6x6 map and 0.005 for the 10x10 map. Without using LLMs the same errors converged to values that were 4 to 6 times larger and more specific 0.035 for the 6x6 map and 0.024 for the 10x10 map. The learned Jacobians projected on the trained 10x10 map are plotted in figure (1). The approximations to the real gradients of the functions show clearly the changing gradient when getting near to the peak of the Gaussian.

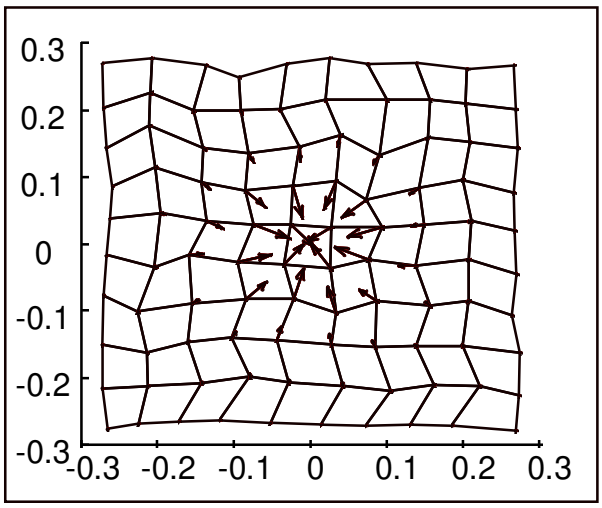

Figure 1. Learned gradients of the function projected on the SOM

IV. System Identification

Another application of the this learning algorithm is in system identification. System identification with neural networks, based on system inputs and outputs, can be performed using a parallel or a series-parallel structure. In a parallel structure, the outputs of the network itself or of the system are fed back to the network through a tapped delay line together with the excitation inputs of the system that are also fed back to the network through a tapped delay line. In a series-parallel structure, the outputs of the real system instead of the outputs of the network are fed back to the network through a tapped delay line. The use of SOMs to cluster concatenated sequential data has already been reported in Kangas [1990]. Here a series-parallel structure is used. It is shown that the expansion around the representative output weights provides the SOM algorithm with the ability to provide a quite accurate smooth estimate of the output compared with the discrete representation provided by the original algorithm.

The input data form vectors of a high dimension that are mapped on a two-dimensional Self-Organizing Map. At the same time the measured outputs are learned together with the Jacobian matrices ($\mathbf{A}_s$).

A. Nonlinear System

The nonlinear system chosen in this example comes from [Narendra et al., 1990] and has the form:

$$y(k+1) = \frac{y(k)}{1 + y(k)^2} + u(k)^3 \qquad (9)$$

Several sizes were used for the moving time windows but the best approximation was given when the size of the tapped delay lines matched the structure of the sytem which means that the best results were obtained with a size of one for both the previous inputs and outputs. Consequently we can say that the system was parametrized by the network in the form

$$y_{net} = \hat{y}\ (k+1) = NN(y(k),u(k)) \qquad (10)$$

where the hat (^) denotes the network's estimate for the system output at time-step t=k+1 as predicted from the network's inputs at step t=k.

The best results were obtained with a 10x10 network that was trained up to 5000 epochs. The training signal was chosen as random in the interval [-2,2]. The initial and final settings of the learning parameters were exactly the same as in the approximation example described above. This network resulted in an RMSE (root mean square error) of 0.24 which, given the range of the outputs, translates in an NRMSE (normalized RMSE) of 0.08 at the end of learning. Further testing with another random signal lying in the same interval and up to 500 epochs resulted in an NRMSE of around 0.09 while without the LLMs one obtains an NRMSE of about 0.38. By plotting this map, the dependence of the map on the cube of the input signal as given from equation (9), is clearly visible. Part of the testing session (figure 3) shows clearly the benefit of the LLMs. The prediction that uses LLMs is hardly discriminable from the actual system output while the standard SOM approximation clearly deviates from the desired values. The normalized RMSE is calculated from the RMSE when it is divided by the standard deviation of the actual system output. That means that if the average of a function is predicted the NRMSE tends to 1 and for perfect prediction it tends to 0.

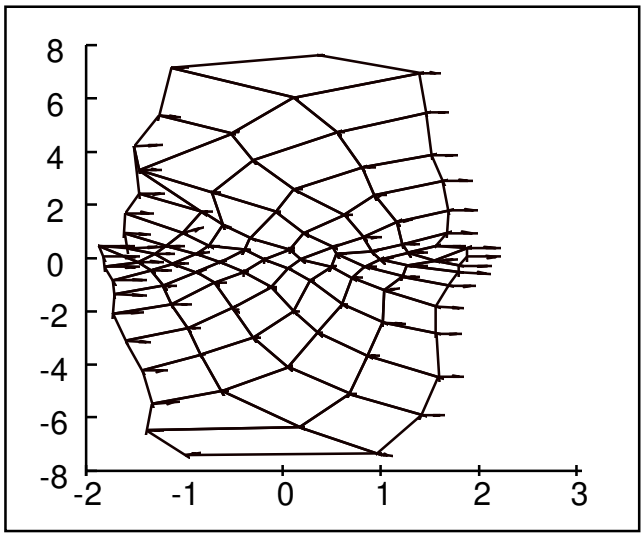

Figure 2. The learned gradients projected on the SOM

B. Experimental

The second system that has been used for testing the above mentioned identification method was an electrohydraulic actuator which consists of a valve with a nonlinear hysteresis characteristic. In such a system, the flow rate (Q), is given as a function of the pressure difference (ΔP) on either side of the valve from a highly nonlinear relation: $Q = k\sqrt{\Delta P}$ where k denotes the orifice flow coeffficient. From this system, input-output data (voltage-to position and voltage to acceleration) have been measured. The excitation in both cases was a random signal lying in the interval [-5,5] Volts and with frequencies between 0 and 20 Hz. A 5x5 map

with LLMs trained for 5000 epochs gave excellent results for the voltage to position data. Testing on another random signal taken from the same interval and with the same frequency content gave an NRMSE of 0.0233, a value which was slightly smaller than at the end of the training session (0.035). For the voltage to acceleration data a 5x5 map trained up to 5000 epochs gave an NRMSE of 0.14 at the end of training. Further testing with a random signal from the same interval and with frequencies up to 20 Hz gave an NRMSE of 0.13. For all the training sessions the initial and final values of the learning parameters were chosen to be equal to the values of the previous examples. The actual output data against the network's estimate for both cases are given in figures 4 and 5.

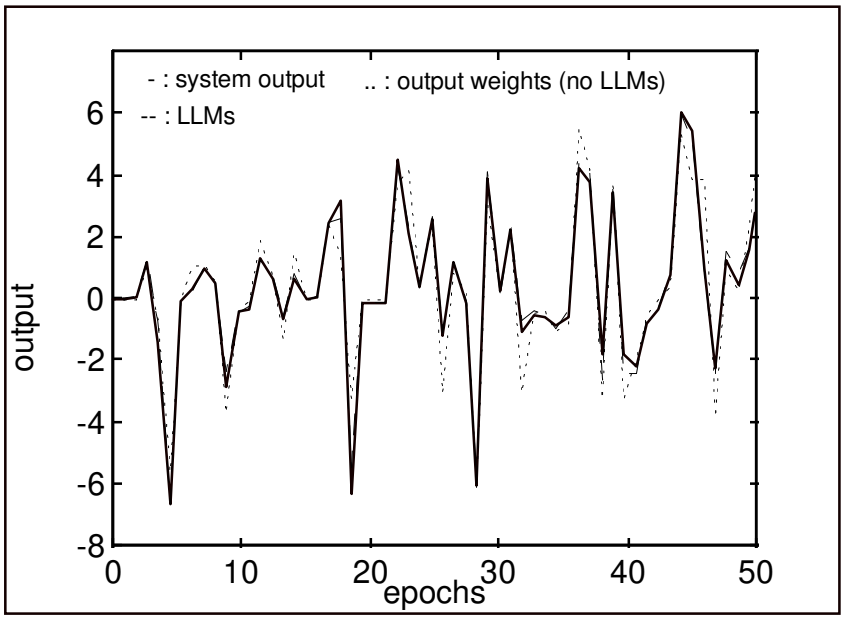

Figure 3. Part of the testing session that shows LLMs vs system output

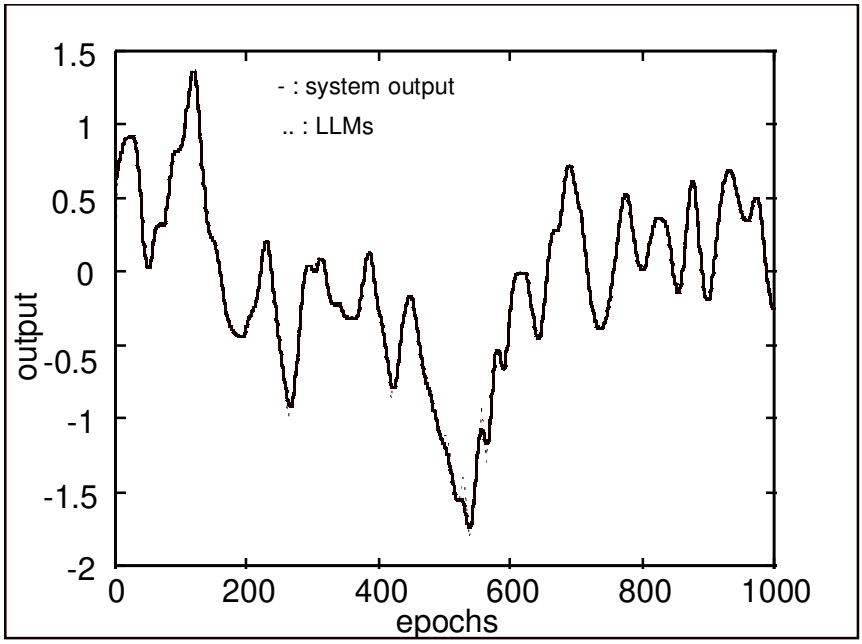

Figure 4. The network's estimate vs the actual position for the electro-hydraulic actuator

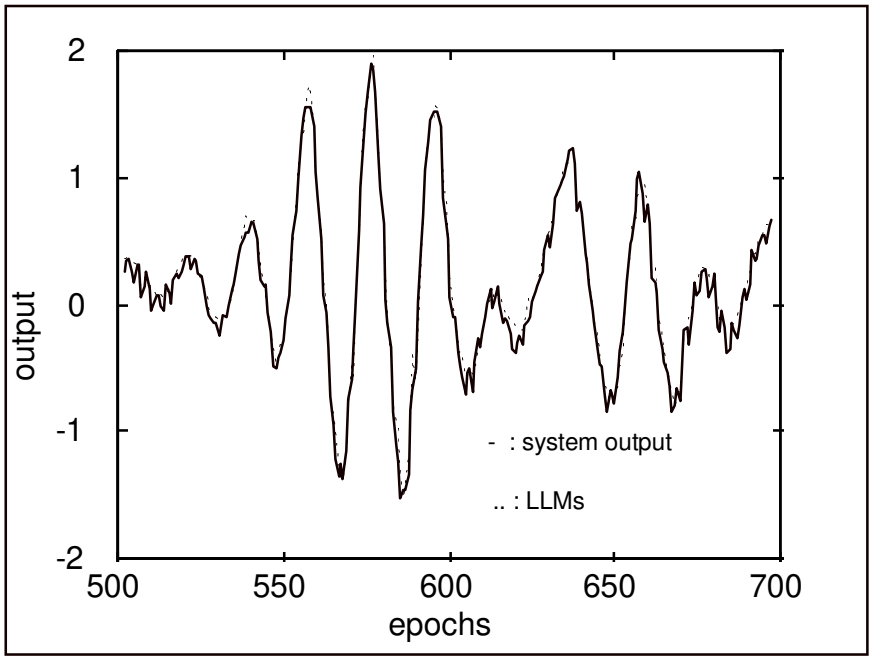

Figure 5. The network's estimate vs the actual acceleration for the electro-hydraulic actuator

V. Conclusions

A new neural network method for function approximation and system identification has been presented. The method has been tested in function approximation and in system identification with promising results. This method can be used equally well for on-line system identification of linear and nonlinear systems or systems with changing parameters because it uses Local Linear Mappings as proven from the results of this paper. Furthermore the gradient information that is stored in the learned Jacobian matrices can be used for several other interpolation schemes (e.g. the construction of membership functions).

References

J. Kangas, (1990) "Time-Delayed Self-Organizing Maps", *Proceedings of the IEEE IJCNN-90 Conference*, S. Diego, CA, vol.2, pp. 331-336

T. Kohonen, (1982) "Self-Organized Formation of Topologically Correct Feature Maps", *Biological Cybernetics*, 43, pp. 59-69

T. Kohonen, (1995) *Self-Organizing Maps,* Springer Series in Information Sciences

T. Martinetz, H. Ritter, and K. Schulten (1990) "Three-Dimensional Neural Net for Learning Visuomotor Coordination of a Robot Arm", *IEEE Trans.on Neural Networks*, vol.1, no.1, pp. 131-136

T. Martinetz, S. Berkovich, and K. Schulten, (1993) "Neural-Gas Network for Vector Quantization and its Application to Time-Series Prediction", *IEEE Trans.on Neural Networks,* vol.4, no.4, pp.558-568

K. Narendra, K. Parthasarathy (1990) "Identification and Control of Dynamical Systems using Neural Networks" *IEEE Transactions on Neural Networks*, vol.1, no.1, pp. 4-27

H. Ritter, T. Martinetz, and K. Schulten (1992). *Neural Computation and Self-Organizing Maps: An Introduction,* Adisson-Wesley, New York. (English and German).

J. Walter, H. Ritter, and K. Schulten (1990) "Non-linear Prediction with Self-Organizing Maps" *Proceedings of the IEEE IJCNN-90 Conference*, S. Diego, CA, vol.3, pp. 589-594