REGULARIZATION VERSUS EARLY STOPPING: A CASE STUDY WITH A REAL SYSTEM

Fernando Morgado Dias[1], Ana Antunes[1] and Alexandre Manuel Mota[2]

[1]Escola Superior de Tecnologia de Setúbal, Instituto Politécnico de Setúbal, Campus do IPS, Estefanilha, 2914-508 Setúbal, Portugal
Tel: +351 265 790000, Fax:+351 265 721869, Email:{ fmdias , aantunes}@est.ips.pt
[2]Departamento de Electrónica e Telecomunicações, Universidade de Aveiro, 3810-193 Aveiro, Portugal
Tel: +351 234 370383, Fax:+351 234 381128, Email: alex@det.ua.pt

Abstract: Regularization and Early Stopping are two of the most common techniques to deal with the overtraining problem in the Artificial Neural Networks field. The overtraining problem appears mostly in systems affected by noise in which after a certain amount of training, the neural network used for modelling starts to learn information specific from the training signal or the noise.
It has already been shown that these techniques can be used to avoid this problem and they are formerly equivalent, but this issue deserve further investigation since real systems sometimes behave in a different way than simulated systems.
A fair comparison for the two techniques is not very easy to make since in the networks there are several parameters that cannot be determined in an analytical way. To overcome this difficulty in the present work a procedure for automating the construction of the models has been used. This procedure allows creating models that are optimised in the number of inputs, the number of hidden neurons and the generalization capability using either Early Stopping or Regularization. This enables the possibility of performing a fair comparison. The procedure includes an hybrid direct/specialized training solution for evaluating the inverse model. To test the results the system used is a reduced scale prototype kiln affected by measurement noise, for which the Direct Inverse Control and Internal Model Control strategies were implemented.

Keywords: Feedforward Neural Networks, Internal Model Control, Measurement Noise, Genetic Algorithm, Overtraining, Early Stopping, Regularization, Direct Training and Specialized Training.

1. INTRODUCTION

The overtraining problem is quite common in the field of Artificial Neural Networks(ANN). This problem is related with the number of training iterations [1] of the network and can be stated as the network learning, after a certain number of iterations, details of the training signal or the noise.
The overtraining problem has been an open topic for discussion motivating the proposal of several techniques like Regularization [2], Early stopping [3] and pruning - Optimal Brain Damage [4] and Optimal Brain Surgeon [5].
The first two are probably the most widely used because of its simplicity to implement and because they do not require too much extra computation.
In [3] it has been shown that Early Stopping is in fact equivalent to Regularization. The distinction is made between implicit regularization (Early Stopping) and explicit regularization.
The issue in this article is in fact whether although formerly equivalent these two techniques attain the same effect in a real system with measurement noise.
This comparison that must be performed is not a straight one since the many parameters that are involved in building the models with or without explicit regularization might hide the true results that are to be compared. To overcome this difficulty in the present work a procedure for automating the construction of the models has been used.

This solution will allow producing models with several degrees of freedom (number of past inputs/outputs, number of hidden neurons and number of iterations or weight decay), using the last parameter to choose between Early Stopping or Regularization.
This procedure that will be shortly introduced is based in a genetic algorithm search in the architecture parameters to find the best model. Using this procedure a fair comparison between Early Stopping and Regularization is provided in order to extract the appropriate conclusions.
The automated procedure includes an hybrid direct/specialized training solution for evaluating the inverse model.
The comparison is made implementing the Direct Inverse Control (DIC) and Internal Model Control (IMC) strategies using as test bench a reduced scale prototype kiln affected by measurement noise.

2. MODELLING ERROR ANALYSIS

The analysis of model quality and error can be found in more depth in [6], [1] and [7], among others and will be presented here in short just to introduce the regularization process.
Suposing a generic model that describes a system y(t) in the following way:

$$y(t) \cong g(\ddot{o}(t),\grave{e}) \quad (1)$$

where ö(t) can be the regressors used in the model and è represents the parameters that can be adjusted in the model.
Assuming that there is a set of input-output pairs of dimension N available:

$$Z_T^N = \{(y(t), j(t)) : t = 1,...,N\} \quad (2)$$

where the T subscript stands for training, since this sequence will be used for training.
The training's objective is to find a minimum of the error between the output of the real system and the output of model.
If the system (and therefore also the training sequence) is affected by noise, then the output can be described by:

$$y(t) = g_0(\ddot{o}(t)) + e(t) \quad (3)$$

where g_0 is the real and unknown system and e(t) is the error.
A measure V(è) of the quality of the model can be of the form:

$$V(q) = E\left\| g_0(j(t)) - g(j(t),q) \right\|^2 \quad (4)$$

According to the structure of the model selected, it is possible to find the best model è*(m), where m represents the dimension of the model:

$$q_*(m) = \arg \frac{\min}{q} V(q) \quad (5)$$

The quality of the model produced can be measured by:

$$EV(\hat{q}_N) = V_*(m) \quad (6)$$

where are the previously estimated parameters.
The result that has been presented by [6], [1] and [7] is about the error obtained by any model:

$$V_*(m) \cong E\left\| g_0(j(t)) - g(j(t),q_*(m)) \right\|^2 + E\left\| g(j(t),q_*(m)) - g(j(t),\hat{q}_N) \right\|^2 \quad (7)$$

The first part of the error is called bias error and the second variance error.
The bias error is due to insufficient model structure and can be decreased by increasing the dimension of the model.
The variance error is due to the fact that the model obtained is not the best attainable within the dimension selected resulting from the noise and limited size of the training sequence.
The fact that the bias error decreases with the increasing of the dimension could lead to the use of very large networks if the variance error would not be expected to increase with the dimension. This is known by bias/variance dilemma [7].
In practice what can be verified is that the training error decreases monotonically with the training iterations while the test error decreases up to a certain point and then starts increasing. This corresponds to overtraining: the ANN starts learning details of the training signal and the noise. This situation makes it possible to understand that the training error continues to decrease while the test error starts to increase.

3. REGULARIZATION AND EARLY STOPPING

For the training algorithms that are based on derivatives the first parameters to be updated are the ones with larger influence in the criteria to be minimized, while in a second phase other less important parameters are updated.
These last parameters to be updated are the ones responsible for the overtraining problem by learning characteristics of the training signal and the noise.
One way to avoid this second phase in training is called regularization and it consists of changing the criteria to be minimized according to:

$$W(q) = V(q) + d\|q\|^2 \quad (8)$$

where ä , the weight decay is a small value.
The idea is to eliminate the so called second phase in learning where parameters with small influence are updated by introducing a trend towards zero in the parameters.
The difficulty is to determine the value of ä, the weight decay, appropriate for performing regularization.
Another way to avoid the overtraining, called early stopping, which is quite intuitive, consists in stopping training before the second phase of training starts but after the first one is concluded so that the characteristics of the system are learned.
Clearly the difficulty here is to find the exact number of iterations for performing the training.
Both solutions have been proved to be formerly equivalent in [3]. Nevertheless it is important that, taking also into account the difficulties to determine the regularization parameter for explicit regularization or the number of iterations to use for

early stopping, to make an evaluation of which solution performs better when tested in a real system with measurement noise.
To make sure that the evaluation is made on the solutions and not on the capacity to determine the parameters, the solutions' quality is measured using the automated procedure that is described in the next two sections.

4. THE AUTOMATED PROCEDURE FOR CREATING THE MODELS

The automated procedure for creating the models is described in detail in [8]. Here this procedure will only be introduced shortly.
The process used for creating and choosing direct and inverse models is based on the same principle, though there are some differences between the two implementations.
There is a common procedure using genetic algorithms for generating a new individual in function of the structure combinations for the Feedforward Neural Networks (FNN) and different ways of evaluating the fitness of the individual.
The structure of the network is composed of four parameters. Three are common: number of past inputs, number of past outputs and number of neurons in the hidden layer and the fourth is the number of training iterations when early stopping is used and for regularization the number of iterations is always the same (equal to the maximum used for early stopping – 256 iterations) and the parameter ä is included.
The choice of the parameters allows some freedom for the models to have more or less complexity according to the modelling necessity and the last parameter allows the choice for implicit or explicit regularization.
In table 1 the information used to code the individuals is resumed indicating the range of values used for each parameter.
The choice of the ranges involves the previous work done with the present system [9], [10].
Each model proposed is an implementation of the structure mentioned above, holding a value for each of the parameters. The models are trained using the Levenberg-Marquardt algorithm because of its fastest convergence.

Table 1 Parameters to use in the optimisation procedure

Parameter	Number of bits	Range of values
Past outputs	2	1:4
Past inputs	2	1:4
Hidden neurons	4	1:16
Iterations / ä	8	1:256 / 1e-5:1e3

The models are all FNN of one hidden layer, with linear output, hyperbolic tangents as hidden layer's activation function and Auto-Regressive with eXogenous input (ARX) architectures.
During the identification and control tasks the NNSYSID [14] and NNCTRL [15] toolboxes for MATLAB were used.
It should be noted that in this automated procedure there is a hybrid direct/specialized training solution for the inverse model. The direct model's fitness is the Mean Square Error (MSE) for the test sequence, while the inverse model's fitness is obtained performing a simulation of DIC.
For additional details please refer to [8].

5. THE FNN STRUCTURE SELECTION USING GENETIC ALGORITHMS

Genetic Algorithm (GA) or Genetic searching algorithm is a function optimisation technique based on the principles of evolutionary genetics and the natural selection process [11] after the pioneering work of Holland [12].
The original goal was to study the adaptation phenomena in nature, but his work was later used for optimisation techniques based on a fitness function, corresponding to the survival of the fittest principle.
Since the initial work many new operators have been proposed and many improvements were introduced but crossover, mutation and elitism are solutions that are present in almost every application of GA for optimisation.
GA optimisation is especially useful when there is no deterministic solution for the problem or the range of solutions is too wide for an exhaustive search and local minimum can be acceptable. It is also important to note that this is a global optimisation method.
The algorithm implemented includes Crossover, Mutation and Elitism the details can be found in [8].
The fitness of the solution is the Mean Square Error obtained between the output of the model and the desired output. The desired output can be the output values in the training set for the direct model or the reference used in the Direct Inverse Control (DIC) simulation for the inverse model.
The fittest solution is the one with the lower fitness value.
A global perspective of the optimisation solution can be obtained from figure 1.
The first block represents the generation of the solutions. The population is represented as a table with binary content, where each line represents an individual of the population and therefore a neural model. The model's parameters are extracted and the model is trained according to the genetic information coded in the individual.
The block called Solution Evaluation represents the fitness evaluation for each solution. If the solution represents a forward model it will be evaluated using a test sequence while if it is an inverse model the evaluation will be done by simulation of control using DIC.
A registry of the fitness of all the population is done in order to select the best elements to compose the elites, which will be used by the genetic algorithm to create the population for the next generation.

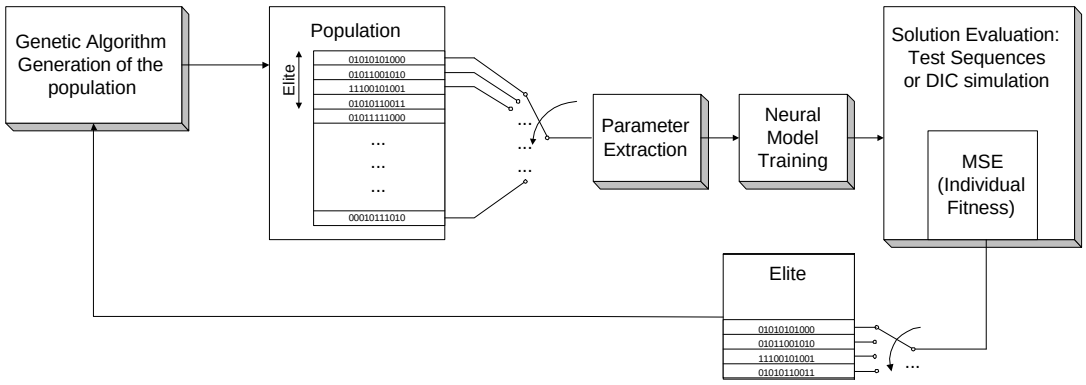

Fig. 1. Representation of the optimisation solution. The block diagram shows the operations performed in each generation of the genetic algorithm.

6. THE PLANT

The plant used to make this comparison is a reduced scale prototype kiln and the tests reported concern the implementation of the temperature control loop.
Figure 2 shows a scheme of the modules composing the system. An electrical resistor driven by a power controller heats the kiln and the temperature is measured by a B type thermocouple. The sensor and the actuator are connected to a Hewlett Packard HP34970A Data Logger that supplies real-time data to MATLAB using the RS232C serial line.

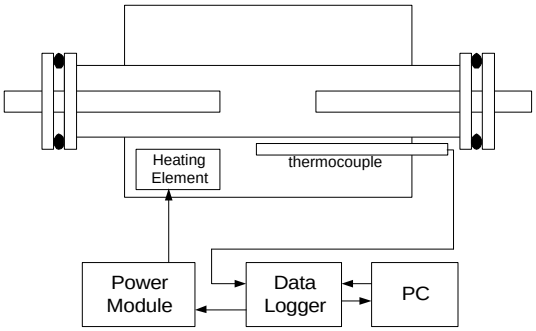

Fig. 2. Schematic of the modules composing the system.

The Data Logger though a helpful tool limits the measurement to temperatures superior to 300ºC and the thermocouple introduces measurement noise, which makes identification more complex. This approach allows the use of the entire MATLAB powerful environment together with real-time capability.
The kiln is completely closed and operates around 750ºC having as superior limit of operation 1000ºC. The Data Logger is used as the interface between PC and the rest of the system. A picture of the system can be seen in figure 3.

7. CONTROL STRUCTURES

The control structures used to test the optimisation procedure are: Direct Inverse Control and Internal Model Control.

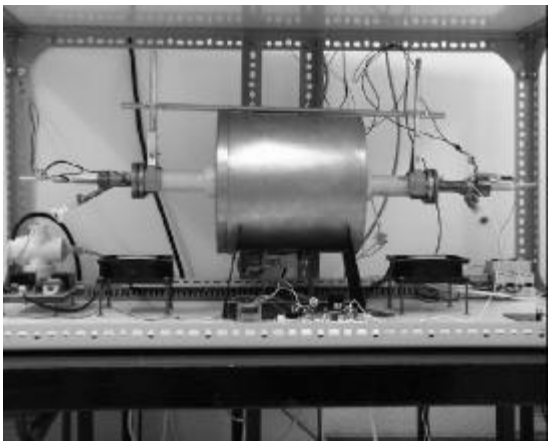

Fig. 3. Picture of the kiln and electronics.

7.1 Direct Inverse Control (DIC).

Direct inverse control is the simplest solution for control that consists of connecting in series the inverse model and the plant as can be seen in figure 4. If the inverse model is accurate the output of the system y(k) will follow the reference r(k).

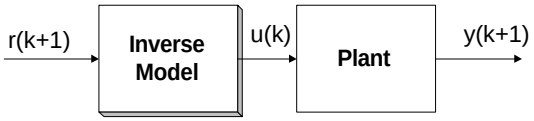

Fig. 4. Structure for Direct Inverse Control. The signal r(k) is the reference, u(k) the control signal and y(k) the output signal.

7.2 Internal Model Control (IMC)

Internal Model Control is a structure that allows the error feedback to reflect the effect of disturbance and plant mismodelling.
In fact it can be shown [10] that a good match between forward and inverse models is enough to have good control and that with this structure disturbance's influence is also reduced. The basic IMC structure can be seen in figure 5.

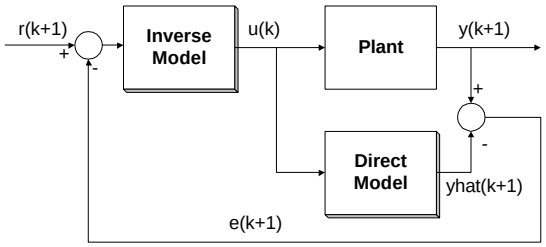

Fig. 5. Structure for Internal Model Control. The signal r(k) is the reference, u(k) the control signal, y(k) the output signal, yhat(k) the estimate of the output and e(k) the error between the output and the estimate.

7.3 Adapting the Control Structures to use Neural Networks Models.

The structure presented in the subsection 7.1 can be used with NN models without the need of major changes, but the structure used in section 7.2 needs some refinements to work properly [13].

The good match between forward and inverse models, referred above translates to having the forward model outputs feedback to the input of the inverse and direct model instead of the outputs of the plant. This means that the inverse model will implement the following equation:

$$u(k) = g\begin{bmatrix} r(k+1), yhat(k),..., yhat(k-n_y+1) \\ u(k-t_d),...,u(k-n_u-t_d+1) \end{bmatrix} \quad (1)$$

instead of:

$$u(k) = g\begin{bmatrix} r(k+1), y(k),..., y(k-n_y+1) \\ u(k-t_d),...,u(k-n_u-t_d+1) \end{bmatrix} \quad (2)$$

Where ny is the number of previous output samples used, nu is number of previous control signal samples used and td is the time delay of the system.

8. THE MODELS OBTAINED

Table 2 shows the details of the best solutions obtained for direct and inverse models, where NU stands for number of past inputs, NY is the number of past inputs, Nhidden is the number of hidden neurons in the hidden layer, Iteration is the number iterations during which the network was trained, ä is the weight decay and NGen is the number of generations used for the NN evolution.

It should be noted that the models are coupled because of the way the inverse models were created. Inverse model 1 was obtained from direct model 1 and so on.

Although this type of combination between Gas and NN is considered to be very slow, in the present case the evolution is very fast when compared with the typical applications. This is due to the fact that the ranges are carefully selected in the beginning and to the efficiency of the LM training algorithm.

Table 2. Parameters of the best models obtained

Model / Parameters	NU	NY	Nhidden	Iteration / ä	Ngen
Early Stopping					
Direct 1	3	3	7	58	49
Inverse 1	1	1	3	225	89
Regularization					
Direct 2	3	4	15	4.475e-4	90
Inverse 2	1	1	13	6.9386	145

9. THE REAL TIME CONTROL ACTION

The models from table 2 were used to implement DIC and IMC according to section V.

The results obtained can be seen in figures 11 to 14 and are summarized in table 3.

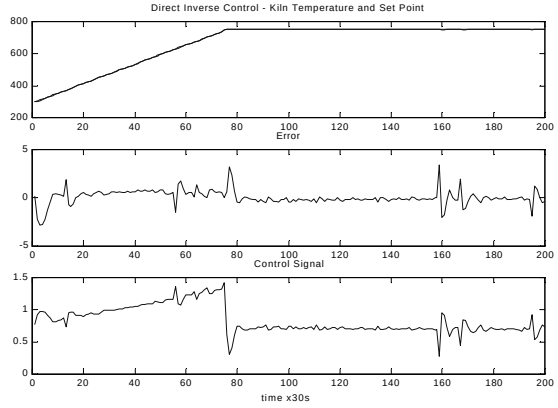

Fig. 6. Direct Inverse Control results with Early Stopping

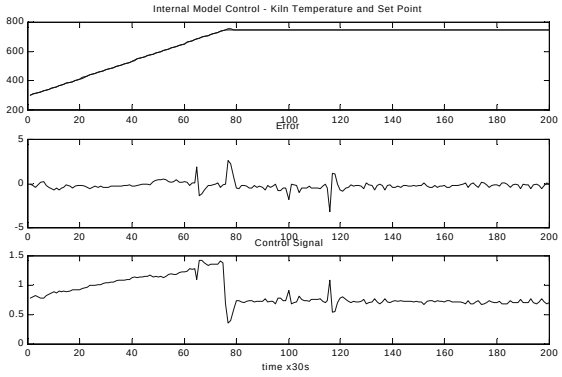

Fig. 7. Internal Model Control results with Early Stopping

Table 3. Summary of the results obtained for both solutions and both control strategies

MODELS\STRATEGY	DIC (MSE)	IMC (MSE)
Early Stopping Models	0.40	0.30
Regularization Models	0.31	0.24

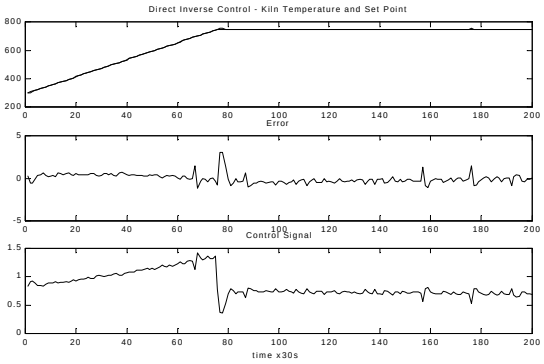

Fig. 8. Direct Inverse Control results with Regularization

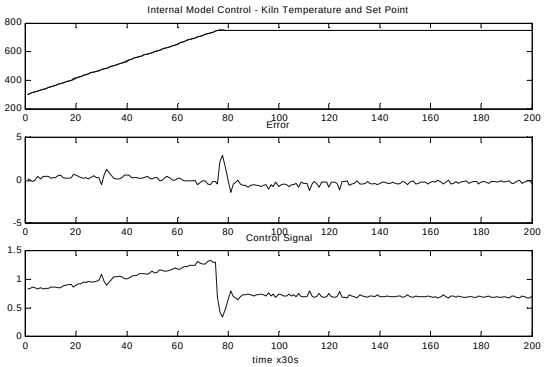

Fig. 9. Internal Model Control results with Regularization

The value of MSE was chosen as a simple measure of performance of the solutions.

As can be seen from table 3, the results obtained with Regularization are slightly better.

10. CONCLUSIONS AND FUTURE WORK

The purpose of the present work was to verify that two techniques that have been proposed and shown to be formerly equivalent for avoiding the overtraining problem attain the same efficacy when dealing with data from a real system with measurement noise.

The results obtained show a superior performance of the Regularization technique of 20% and it can also be added that the models obtained this way are, in this study, more consistent.

Nevertheless it should be noted that:

- The Early Stopping technique is a valid solution since it is simpler to use when the models are created without an automated procedure.
- The models obtained with Regularization are more complex than the ones obtained with Early Stopping. As a consequence the models obtained with Regularization will have longer execution times.

The use of the Levenberg-Marquardt algorithm and the correct choice of the ranges for the parameters allowed achieving short time ANN evolution.

As future work the fitness function can include an additional part to add a penalization in function of the complexity of the ANN obtained. This will avoid obtaining a complex model that is just slightly better than another simpler model.

REFERENCES

[1] J. Sjöberg, "Non-linear System Identification with Neural Networks", PhD Thesis, Dept. of Electrical Engineering, Linköping University, Sweden, 1995.

[2] M. W. Pedersen and L.K. Hansen, “Recurrent Networks: Second Order Properties and Pruning”, Proceedings of Neural Information Processing Systems 7, 673:680, 1994.

[3] J. Sjöberg and L. Ljung, “Overtraining, Regularization and Searching for minimum in Neural Networks”, Preprint IFAC Symp. on Adaptive Systems in Control and Signal Processing, Grenoble, France, 669:674, 1992.

[4] Y. Le Cun, J.S. Denker and S.A. Solla, “Optimal Brain Damage, Advances in Neural Information Processing Systems”, Denver 1989, Ed. D.Touretzsky, Morgan Kaufmann, 598:605, 1990.

[5] B. Hassibi and D. G. Stork, “Second Order Derivatives for Network Pruning: Optimal Brain Surgeon”, Proceedings of NIPS 5,164, San Mateo, California, 164:172, 1993.

[6] L. Ljung, "System Identification - Theory for the User", Edited by Prentice Hall, 1987.

[7] J. Sjöberg and L. Ljung, “Overtraining, Regularization and Searching for minimum in Neural Networks”, Preprint IFAC Symp. on Adaptive Systems in Control and Signal Processing, Grenoble, France, 669:674, 1992.

[8] F. M. Dias, A. Antunes and A. M. Mota, “Automating the Construction of Neural Models for Control Purposes using Genetic Algorithms”, 28th Annual Conference of the IEEE Industrial Electronics Society, Sevilla, Spain, 2002.

[9] F. M. Dias and A. M. Mota, “A Comparison between a PID and Internal Model Control using Neural Networks”, 5th World Multi-Conference on Systemics, Cybernetics and Informatics, Orlando, EUA, 2001.

[10] F. M. Dias and A. M. Mota, ”Comparison between different Control Strategies using Neural Networks”, 9th Mediterranean Conference on Control and Automation, Dubrovnik, Croatia, 2001.

[11] J.-S. Yang and M. L. West, “A Case Study of PID Controller tuning by Genetic Algorithm”, Proceedings of the IASTED International Conference on Modelling Identification and Control, Innsbruck, 2001.

[12] J. Holland, “Adaptation in Natural and Artificial Systems”, University of Michigan Press, Ann Arbor, MI, 1975; MIT Press, Cambridge, MA, 1992

[13] G. Lightbody and G. W. Irwin, “Nonlinear Control Structures Based on Embedded Neural System Models”, IEEE transactions on Neural Networks, vol.8, no.3, 1997.

[14] M. Nørgaard, “Neural Network System Identification Toolbox for MATLAB”, Technical Report,1996.

[15] M. Nørgaard, Neural Network Control Toolbox for MATLAB, Technical Report,1996.

[16] K. J. Hunt, D. Sbarbaro, R. Zbikowski and P. J. Gawthrop, ”Neural Networks for Control Systems-A Survey”, Automatica, vol.28, nº6, pp1083-1112, 1992

[17] O. Sørensen, Neural Networks in Control Applications. PhD Thesis, Department of Control Engineering, Institute of Electronic Systems, Aalborg University, Denmark, 1994.

[18] M. Nørgaard, “System Identification and Control with Neural Networks”. PhD Thesis, Department of Automation, Technical University of Denmark, 1996.