Proceedings of the IV Brazilian Conference on Neural Networks - IV Congresso Brasileiro de Redes Neurais
pp. 019-024, July 20-22, 1999 - ITA, São José dos Campos - SP - Brazil

A NEURAL NETWORK APPROACH IN A BACKWARD HEAT CONDUCTION PROBLEM

Fabio Tokio Mikki[1], Edison Issamoto[1], Jefferson I. da Luz[1],
Pedro Paulo Balbi de Oliveira[1], Haroldo F. Campos-Velho[2],
Jose Demisio Simoes da Silva[2]
[1]Universidade do Vale do Paraíba – IP&D-UNIVAP
[2]Instituto Nacional de Pesquisas Espaciais – LAC-INPE
Emails: pedrob@univap.br, {demisio, haroldo}@lac.inpe.br

Abstract

This paper describes the experiments conducted in determining the initial temperature distribution on a slab with adiabatic boundary conditions, from a transient temperature distribution, obtained at a given time. This is an ill-posed inverse problem, where the initial condition has to be estimated. Two different artificial neural networks have been applied to address the problem: backpropagation and radial basis functions (RBF). Both approaches use the whole temperature history mapping. In our simulations, RBF presented better solutions, faster training, but higher noise sensitiveness, as compared to backpropagation.

1. Introduction

This paper describes a neural network based approach to the problem of determining the initial temperature distribution on a slab with adiabatic boundary conditions, from a transient temperature distribution, obtained at a given time. The problem is considered to be an ill-posed inverse problem, where the initial condition has to be estimated [4]. Two neural network (NN) models were used: a feedforward network with backpropagation and a radial basis function (RBF) network, both of which differ in topology and use different training strategies. Our simulations have proved their effectiveness for solving the inverse problem at hand, under the problem approach of using the whole temperature history mapping (WHM) for training purposes. Comparisons of simulations with both models have shown a better performance of the RBF network with Gaussian functions, because of faster training and better solutions to the problem.

2. The Inverse Problem

The direct problem consists of a transient heat conduction problem in a slab with adiabatic boundary condition with an initial temperature distribution denoted by $f(x)$. The mathematical formulation of this problem is given by equation (1) below

$$\frac{\partial^2 T(x,t)}{\partial x^2} = \frac{\partial T(x,t)}{\partial t}, \quad \text{for } (x,t) \in \Omega \times \mathrm{R}^+,$$
$$\frac{\partial T(x,t)}{\partial x} = 0, \quad (x,t) \in \partial\Omega \times \mathrm{R}^+, \tag{1}$$
$$T(x,0) = f(x), \quad (x,t) \in \Omega \times \{0\}$$

where $T(x,t)$ (temperature), $f(x)$ (initial condition), x (spatial variable) and t (time variable) are dimensionless quantities and $\Omega = [0,1]$.

The solution to the direct problem for a given initial condition $f(x)$ is explicitly obtained using separation of variables [4], for $(x,t) \in \Omega\times\mathrm{R}^+$:

$$T(x,t) = \sum_{m=0}^{+\infty} e^{-\beta_m^2 t} \frac{1}{N(\beta_m)} X(\beta_m, x) \int_0^1 X(\beta_m, x') f(x') dx' \tag{2}$$

where $X(\beta_m,x) = \cos(\beta_m x)$ are the *eigenfunctions* associated to the problem, $\beta_m = m\pi$ are the *eigenvalues* and $N(\beta_m) = \int_\Omega X(\beta_m, x') f(x') dx'$ represents the *integral normalization* (or the *norm*).

This paper presents a method to obtain the inverse solution using a neural network approach, whose architecture makes use of two neural networks as it is shown by the block diagram in figure 1:

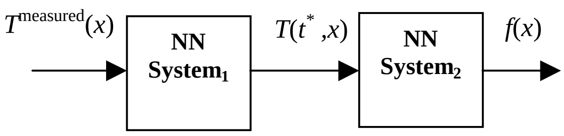


Figure 1 – Block Diagram of the NN approach.

3. Backpropagation and RBF Neural Networks

Neural networks have emerged from an obscure field, discredited by perceived inadequacies into one of the fastest growing technologies in information processing [10] and [1]. Much research has being done in pursuing new neural network models and adapting the existing ones to solve real life problems, such as those in engineering [10] and [1].

The basic component of a NN is a neuron, which was modeled by McCulloch and Pitts in 1943 [1] as a computational model of a biological neuron. The arrangements of such processing units make the NNs, which are characterized by:

- A large number of very simple neuronlike processing elements.
- A large number of weighted connections between the elements that encode the knowledge of a network.
- Highly parallel, distributed control.
- An emphasis on learning internal representations automatically.

The basic idea is that a massively parallel network of simple elements can arrive at a result very fast and, at the same time, display insensitivity to the loss and the failure of some number of component elements in the network [12]. These important properties make neural networks appropriate for applications such as pattern recognition, signal processing, image processing, financing, computer vision, engineering, etc. [1], [10], [12], and [9].

The processing element in a NN is a linear combiner with multiple weighted inputs, followed by an activation function. The simplest NN is the Perceptron that has a hard limiter activation function, being appropriate for solving linear problems.

There are several different architectures of NNs, most of which directly depend on the learning strategy adopted. It is not the aim of this paper to go on a detailed background on NNs. Instead, we will concentrate on a brief description of the two NNs used in our simulations: the multilayer perceptron with backpropagation learning and radial basis functions (RBF). Good introductions on NNs can be found in [12] and [10].

The multilayer perceptron with the backpropagation learning algorithm, also referred to as the backpropagation neural network is a feedforward network composed of an input layer, an output layer, and a number of hidden layers, whose aim is to extract high order statistics from the input data [12].

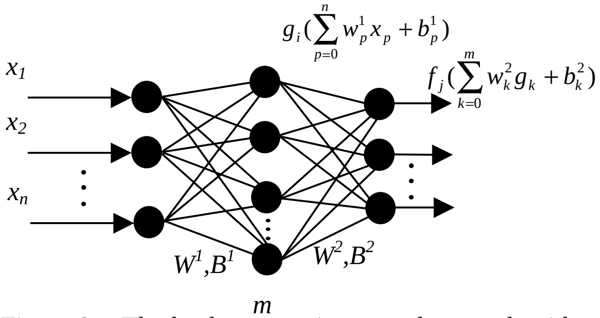


Figure 2 – The backpropagation neural network with a hidden layer.

Figure 2 shows a backpropagation neural network with only one hidden layer. g and f are activation functions for the neurons in the hidden layer and in the output layer respectively. In order to introduce more flexibility to the network to solve non-linear problems, the activation functions for the hidden layer are sigmoid functions $f(x)=\frac{1}{1+e^{-x}}$, varying between 0 and 1, or hyperbolic tangent functions $f(x)=\frac{1-e^{-x}}{1+e^{-x}}$, which varies between –1 to 1.

In a backpropagation network, a supervised learning algorithm controls the training phase. Then, the input and output (desired) data need to be provided, thus permitting the calculation of the error of the network as the difference between the calculated output and the desired vector. The network's weights adjustment is conducted by backpropagating such error to the network. The weight change rule is a development of the perceptron learning rule. Weights are changed by an amount proportional to the error at that unit times the output of the unit feeding into the weight. Equation (3) shows the general weight correction for the delta rule.

$$\Delta w_{ji} = \eta \delta_j y_i \qquad (3)$$

δ_j is the local gradient, y_i is the input signal of neuron j, and η is the learning rate parameter that controls the strength of change.

Radial basis function networks are feedforward networks with one hidden layer, developed for data interpolation in multidimensional space [7]. Like backpropagation networks, RBFs can learn arbitrary mappings. The difference between the two networks is that RBF hidden layer units have a receptive field with a center, through which a particular input value has a maximal output. Their output tails off as the input moves away from this point. Generally, the RBF hidden unit function is a Gaussian function with null mean and standard deviation σ^2 as it is shown in (figure 3). Figure 4 shows an RBF network.

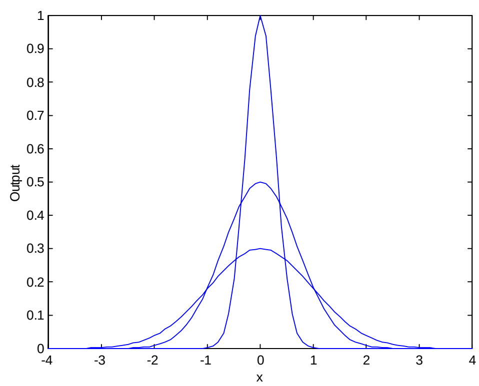


Figure 3 – Gaussians with three different standard deviations.

The training of RBFs, consists in deciding the number of hidden units there should be, the centers and the sharpness (standard deviation) of their Gaussians, and then training up the output layer. Generally, the centers and standard deviations are decided on first by examining the vectors in the training data. The output layer weights are then trained using the Delta rule.

RBF networks can be used for classification problems and function approximation. They have the advantage that one can add extra units with centers near parts of the input, which are difficult to classify.

Both Backpropagation and RBFs networks can be used for processing time-varying data and many other applications.

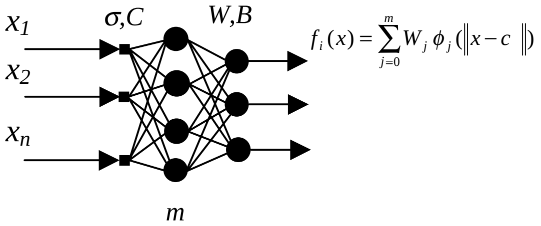


Figure 4 – Radial Basis Functions network.

4. Implementation

The problem approach adopted for the inverse problem in this paper is based on the Whole History Mapping (WHM) presented in [3]. The basic idea is to design a NN for mapping the whole vector of observed values of the temperature history to the corresponding vector of outputs (the heat transfer coefficients in [3]). After training, there is an exact correspondence between the observed temperature values and the output, over some time interval.

The WHM has the advantage of being quite stable and insensitive to noise in the data [1]. However, two critical disadvantages are: 1) the number of input vectors can be quite big, which leads to a large number of connections and very slow training; and 2) for every case a new network has to be constructed, thus requiring new training sets, and the whole training process.

In our simulations, three temperature distributions were obtained using the direct model of [4] over a slab with adiabatic conditions (figure 4), for 50 time slices.

Two experiments were conducted using two neural network arrangements: two backpropagation networks and two RBF networks. The aim was to find out the differences between the two models in estimating the initial temperature distribution. The general architecture of both arrangements is presented in figure 7. Network 1 is used for estimating the time of the corresponding measured temperature distribution (only 1 output neuron is required) and network 2 is used for the iterative estimation of the initial temperature distribution over the slab (*n* output neurons representing different positions over the slab).

The network architectures used in both experiments had the following features:

- 2 Backpropagation networks:
  - 1 input layer; 1 hidden layer; output layer
  - Hyperbolic Tangent hidden neurons
  - Linear output neurons
- 2 RBF networks:
  - 1 input layer; 1 hidden layer; 1 output layer
  - Gaussians Radial basis neurons
  - Linear output neurons Linear

The training data consisted of the appended distribution of three temperature distributions obtained through the direct model (2) shown in figure 5. The three distributions had different initial profiles: triangular, logarithmic, and sinusoidal, with maximum value 1.

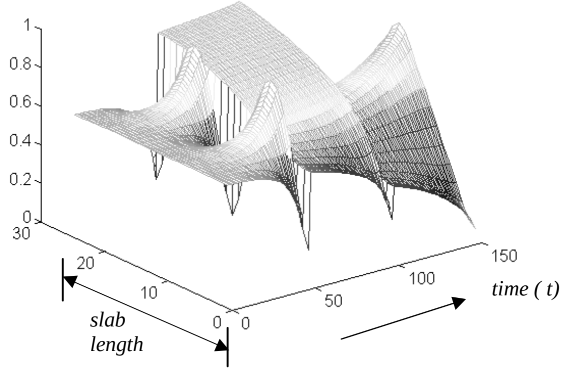


Figure 5 – Combination of three temperature distributions over 25 positions over the slab, for 50 time slices.

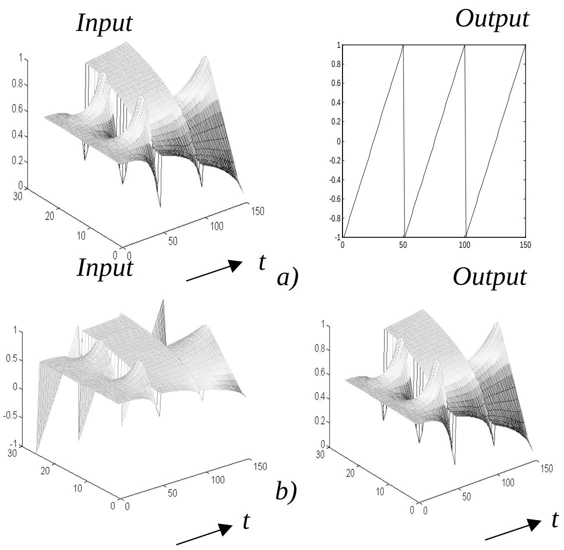


Figure 6 – Data sets used for training the neural networks. a) Input and Output data sets for network 1; b) Input and Output data sets for network 2.

Two distinct training sets were derived from the resulting data in figure 5: one for training network 1 (figure 7) that estimates the time slice corresponding to a certain temperature distribution; and another for training network 2 (figure 7) that calculates the initial temperature distribution over the slab (figure 6). The input data for network 1 was the temperature distribution history (figure 6-a left) and target data were the time slices corresponding to each temperature distribution over the slab (figure 6-a right). For network 2 the input was the time appended temperature distribution (figure 6-b left) and target data was the shifted temperature distribution (figure 6-b right).

Table 1 – Training results for the backpropagation neural networks.

network	# of neurons	Target error	# of epochs
1	50	0.001	20000
2	50	0.0001	20000

Hyperbolic tangent hidden neurons
Linear output neurons

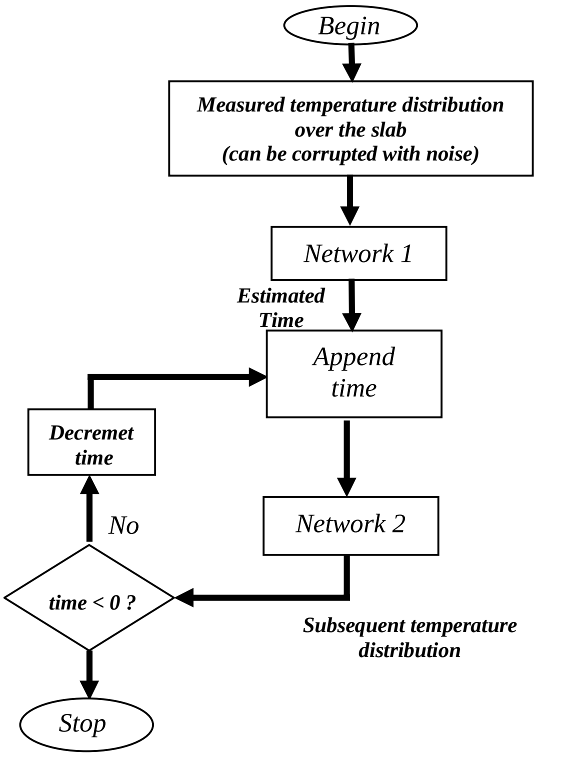


Figure 7 – Flowchart of the general approach for applying neural networks to solve the inverse problem described in [4].

For network 2, in both simulations, the training consisted in presenting a measured temperature profile with its corresponding time slice (t) at the input, and the following distribution in the time scale (t+1) as the output. Then, network 2 learned how to estimate the distribution at a time slice ahead, whereas network 1 estimated the time of occurrence of the measured temperature distribution supplied.

The training was performed separately for each network and each arrangement. Table 1 shows the training results for the backpropagation networks, together with the parameters used for training. Both networks were trained up to the stopping criteria: network 1 achieved the target error, whereas network 2 was trained for the maximum number of epochs. Simulations were conducted under the Matlab neural network toolbox.

The training of the RBF networks used similar parameters as it is shown in table 2. The algorithm for training the RBF networks gradually builds up the final network, thus not requiring the specification of the number of neurons in the hidden layer.

In both simulations, the training parameters were chosen testing different possibilities and checking out the networks' performances.

Table 2 – Training results for the RBF networks.

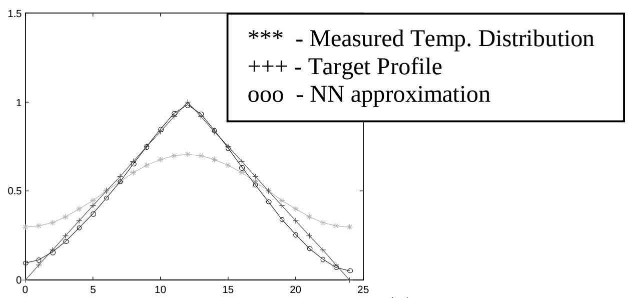
network	RBF function	# of neurons	Spread constant	Target error
1	Gaussian	34	0.1	5×10^{-3}
2	Gaussian	59	0.3	10^{-5}

The performances were tested through the activation of both arrangements by presenting a certain temperature distribution over the slab to network 1, chosen from one of the individual distributions that make up the one in figure 5. Network 1 estimated the time of the given distribution, which was fed to network 2, together with the given distribution, for iterative estimation of the initial distribution. Figure 7 shows the flowchart for activation of both networks for each simulation. It is important to notice that both networks were trained with the combined data sets.

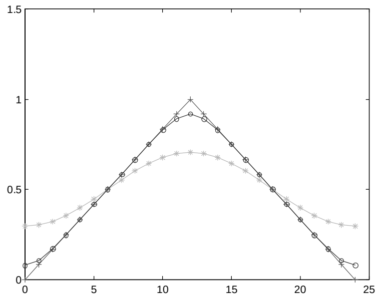
5. Results

Figure 8 presents results of the activation of the networks trained with the combined distribution in figure 6. The activation consisted in presenting the networks with a certain profile chosen from one of three individual distributions that make up the combined distribution. The profiles supplied to the networks were chosen at four different positions on the individual distributions, corresponding to $^1/_2$, $^1/_5$, $^1/_{10}$, and $^1/_{25}$ slices of the total time for steady state.

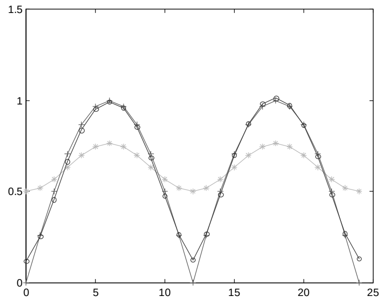
In figure 8, the plots show some of the results obtained with the activation of the backpropagation networks and the RBF networks with the same measured temperature distribution. The curves with (*) represent the measured temperature distribution supplied to the networks. The curves with (+) represent the target profile, that is, the desired initial temperature distribution. The curves with (o) represent the NN approximation. In figures 8-a e 8-b show the backpropagation and RBF results for a triangular distribution chosen at ½ of the time of steady state. It is



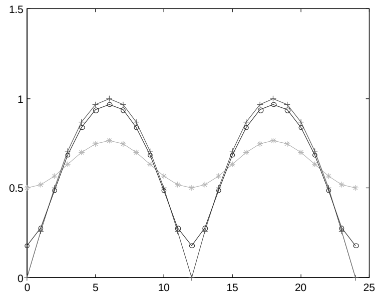
(a)



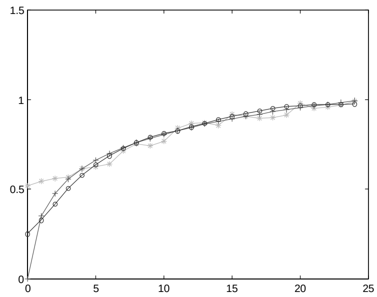
(b)



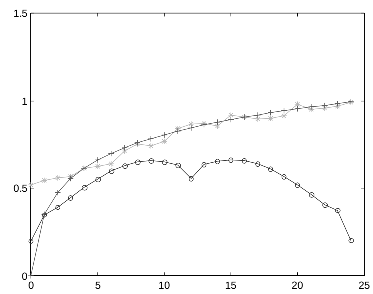
(c)



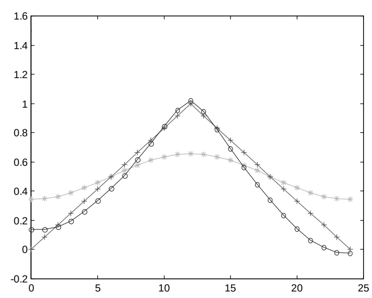
(d)



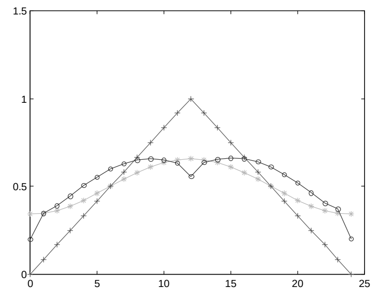
(e)



(f)



(g)



(h)

Figure 8 – Results of activation of both network arrangements. (***) - Measured Temp. Distribution; (+++) - Target Profile; (ooo) - NN Approximation.

to be notice that the backpropagation network approximated better to the desired profile. Figures 8-c and 8-d show the results of the backprogation and the RBF networks for a temperature distribution measured at $^1/_5$ of steady state time. In figures 8-e and 8-f, a logarithmic distribution was provided chosen at $^1/_{25}$ of steady state time with noise added at a 5% rate. The perturbation was constructed with a vector of uniform random numbers weighted by a certain percentage (5%). Figures 8-g and 8-h show the results for the backpopagation and RBF networks for a distribution, which was not used for training them. Comparison shows the backpropagation networks generalize better than RBF networks. Table 3 summarizes the activation results for different initial profiles and different time instances, some of them perturbed with some noise.

Table 3 – Summary of activation results

Distribution	Time	Error Back	Error RBF	Noise
Triangular	½	0.001932	0.000912	0%
Triangular	$^1/_5$	0.000573	0.000425	0%
Triangular	$^1/_{25}$	0.000706	0.000408	0%
Sinusoidal	$^1/_5$	0.002148	0.004418	0%
Logarithmic	$^1/_{25}$	0.003241	0.001665	0%
Triangular	$^1/_5$	0.001311	0.000465	5%
Logarithmic	½	0.002974	0.102762	5%
Triangular	$^3/_{10}$	0.009997	0.039955	0%

6. Conclusions

This paper has presented a method for solving an inverse initial condition problem in heat conduction, using a neural network approach. The problem was approached using two NN models: backpropagation and RBF networks. The training sets were constructed with the direct model in equation 2. Two NN systems were used composed of two backpropagation networks or two RBF networks.

The results (figure 8 and table 3) show backpropagation networks interpolate better for unknown data (figure 8-g) and are more robust to perturbations in the data (figure 8-e), whereas RBF networks train faster and do not require architecture specification beforehand but are very sensitive to noise (figure 8-h).

The experiments conducted show the effectiveness of neural networks in solving inverse problems. The results in figure 8 and table 3, were obtained in the experiment of an on going research on using neural networks for solving inverse problems. Although they prove the effectiveness of NN in inverse problem, future work still has to be done. In these regards, the research will continue in trying to establish numerical comparisons with those of [4]. Also, different NN architectures and more diverse training sets will be tried in searching new neural network architectures and checking generalization of the models. In addition, the proposed methodology will be tried to solve inverse problems in other application areas such as geophysics, image processing and computer vision.

Acknowledgements

This work has been funded by Fundação de Amparo à Pesquisa do Estado de São Paulo - FAPESP, (Thematic Project process 96/07200-8), and the British Council (Ref. SPA/881/126).

References

[1]Haykin, S., Neural Networks Expand SP´s Horizons. IEEE Signal Processing, March 1996, pp. 24 – 49.

[2] Widrow, B.; Winter, R., Neural Nets for Adaptive Filtering and Adaptive Pattern Recognition. Computer, 21(3), March 1988, pp. 25-39.

[3]Krejsa, J.; Woodbury, K.A.; Raudensky, M., Assessment of Strategies and Potential for Neural Networks in the IHCP. 1996.

[4]Muniz, W.B.; Campos Velho, H.F.; Ramos, F.M., A Comparison of Some Inverse Methods for Estimating the Initial Condition of the Heat Equation. Journal of Computational and Applied Mathematics, Vol. 101, pp. 153-171, 1999.

[5]Musavi, M.T.; Ahmed, W.; Chan, K.H.; Faris, J.B.; Hummels, D.M., On the Training of Radial Basis Function Classifiers. Neural Networks, Vol. 5, pp. 595-603, 1992.

[6]Chen, S.; Cowan, C.F.N.; Grant, P.M., Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks. IEEE Transactions on Neural Networks, vol. 2, No. 2, pp. 302-309, March 1991.

[7]Mulgrew, B., Applying Radial Basis Functions. IEEE Signal Processing Magazine, pp. 50-65, 1996.

[8]Krejsa, J.; Sláma, L.; Horský, J.; Raudenský, M.; Pátíková, B., The Comparison of the Traditional and Non-classical Methods Solving the Inverse Heat Conduction Problem. 1996.

[9]Lin, C-T; Lee, G. Neural Fuzzy Systems: A Neuro-Fuzzy synergism to Intelligent Systems. Prentice Hall, New Jersey, 1996.

[10]Tsoukalas, L.H.; Uhrig, R.E., Fuzzy and Neural Approaches in Engineering. John Wiley, New York, 1997.

[11]Haykin, S. Neural Networks: A Comprehensive Foundation. Macmillan, New York, 1994.

[12]Nadler, M.; Smith, E.P., Pattern Recognition Engineering. John Wiley, New York, 1993.

[13]W.B. Muniz, F.M. Ramos, H.F. Campos Velho (1999): Solution of the Backward Heat Conduction Equation Using Different Regularization Techniques, To appear in: Computers & Mathematics with Applications.