Transactions on Information and Communications Technologies vol 2, © 1993 WIT Press, www.witpress.com, ISSN 1743-3517

Neural network, expert system and inverse problems

V. Dumek, M. Druckmüller & M. Raudensky
Technical University of Brno, Technicka 2, 616 69 Brno, Czech Republic

Inverse problems are problems of determining cause on the basis of the knowledge of their effects. The object of the inverse heat conduction problem is to determine the external heat transfer (the cause) given observation of the temperature history at one or more interior points (the effect).

This communication shows a new approach to inverse problems. This approach uses a neural network and an expert system. The examples shown in this paper were computed using back propagation software (neural network) and a system based on Lukaszewicz's many valued logic (expert system). The numerical technique of neural networks evolved from the effort to model the function of the human brain and expert systems take the place of expert knowledge in several areas.

The solution to an inverse problem by neural network and the same by expert system can be divided into two parts:

For neural networks:

- *training (when the weights of the connections among the processing elements are modified)*
- *evaluation (when the trained network responds to input data).*

For expert systems:

- *creation of a knowledge base (pieces of knowledge are put into the base).*
- *answer of the system (when the system gives answers to questions).*

1. Inverse Task

The inverse heat conduction problem is a problem of determining the external heat transfer (the cause) given observation of the temperature history at one or more interior points (the effects). This contrasts with the usual forward problem in heat conduction, which is to determine the internal temperature field for a given set of boundary conditions.

The one dimensional direct problem is equal to the solution by the partial differential equation:

$$\frac{\partial^2 T}{\partial x^2} = \frac{1}{a}\frac{\partial T}{\partial t} \tag{1}$$

where T is a set of temperatures, x of spatial coordinates, a for thermal diffusivity and t for time. Boundary conditions must be known for the solution of the equation (1). These boundary conditions are in our case described by ambient temperature and the heat transfer coefficient at the boundary of the body. The computation of temperature fields from equation (1) is a routine problem. The finite difference method or finite element method are frequently used. The problem becomes more complex when an inner temperature is known (for example from experiment) and boundary conditions need to be found. This is an inverse problem. This problem is from a mathematical point of view incorrect and usually from a numerical point of view ill-posed.

Beck's inverse method of sequential function specification is well-established for one-dimensional problems [1, 2]. The essence of this method is that a functional form for the unknown variation in the external heat transfer is presumed; typically a piecewise constant or piecewise linear variation is assumed. The remaining task in the method is to determine the unknown parameters in the assumed functional form which will minimize the sum of the error between the observed and computed values of temperature at the measurement points. Usually, a sum-squared-error performance measure is used, hence an least-squares minimization routine is generally used.

2. Neural Network

It is not the aim of this paper to describe the inherent problems of the neural network technique. Thus the neural network concept will be mentioned only briefly.

A neural network is a structure for parallel information processing. The network is loaded by the input signal

$$\mathbf{X} = (x_1, x_2, \ldots, x_n),$$

and the network responds with the output signal

$$\mathbf{Y} = (y_1, y_2, \ldots, y_m).$$

The transformation of the signal can be written in the form:

$$\mathbf{Y} = f\,(\mathbf{X}, \mathbf{W}),$$

where **W** is the vector of weights. Weights are the dynamic structure of the neural network that are adjusted to perform a required task. Each element in the network receives as its input the output of the previous element to which it is attached, multiplied by the weight value on the connection between the two elements.

The structure consists of a number of processing elements. Each processing element can receive in parallel a finite amount of input information. The processing element transforms the input data with an activation function. The output signal can be

Transactions on Information and Communications Technologies vol 2, © 1993 WIT Press, www.witpress.com, ISSN 1743-3517

received by an arbitrary number of processing elements. The schematic structure of a processing element is shown in Fig.1.

The input to this element is the sum of the activations of the preceding elements to which it is connected, multiplied by the weight value of each connection. This can be written in the form:

$$I_j = \sum_{i=1}^{n} W_{i,j} X_i$$

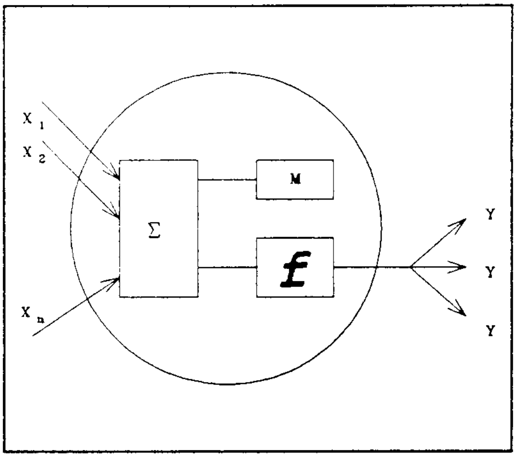

Fig.1 Processing element

where I_j is the input to the element j, W_{ij} is the weight value of the connection from the element i to element j and X_i is the output from the previous element i. The output of the element j is given by transforming the input I_j by the activation function.

Any activation function can be used with the back-propagation network. The only restriction is that there exists the first derivative. In the following examples a sigmoidal function was used:

$$f = \frac{1}{(1 + e^{-I})}$$

2.1 Organization

The study of neural network simulation originates from the effort to understand the process of thinking in the human brain. The structure and ability of the human brain and neural network, which is used for the present computations, can hardly be compared.

Even if the human brain is extremely complicated it can be stated that the overwhelming structure is layered.

The elements and connections which form a back-propagation neural network are organised into layers. Fig.2 shows the commonly used structure. The structure consists of one input layer, one output layer and several intermediate (hidden) layers. The input layer has one element for each element of the input data pattern. The input layer processing elements only distribute the input signal to the elements of the first hidden layer. The output layer has one element for each of the desired outputs.

In dealing with our problem there are two distinct phases in the use of the back propagation network: **training and application.** In the first phase, the network is shown input/output pairs from the system which the user wishes to simulate. The input patterns are represented by a set of **X** vectors, and the output by a set of **Y** vectors. A process involving a learning algorithm takes place, in which the weights of the network

Transactions on Information and Communications Technologies vol 2, © 1993 WIT Press, www.witpress.com, ISSN 1743-3517

connections are adjusted with each pair of **X, Y** vectors so as to better reproduce the desired behaviour. In the second phase, the trained network is implemented through software to perform the desired task.

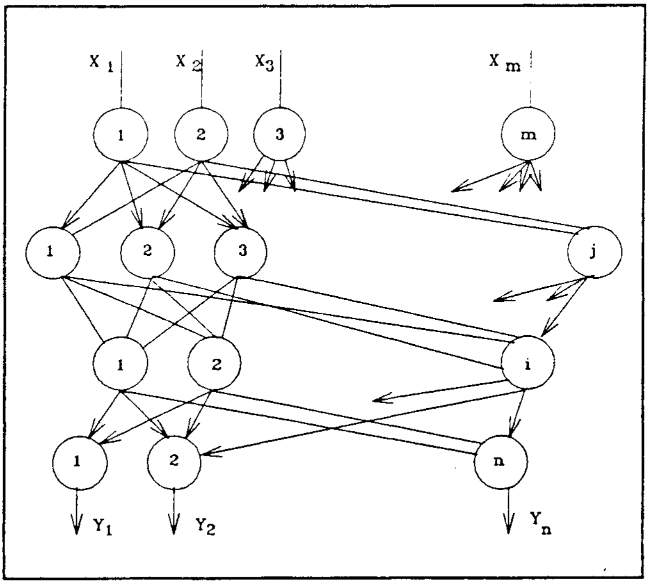

Fig. 2 Neural network structure

The network initially begins with random weight values connecting the various elements. A set of values from the input data file is assigned to the input units. These values are propagated forward through the network by summing element inputs and calculating element outputs from the first layer to the last. At the output layer the calculated values are compared with the desired values. The difference is then propagated back toward the input units as the error signal. As this signal reaches each individual connection in the network the weight is modified so as to reduce the overall modelling error.

The process of fixing input/output values and changing weights is repeated for a number of data sets. With each iteration the weights are modified so that the network more closely produces the correct output from the given input. The learning process is continued until the prescribed accuracy of response or a prescribed number or learning runs is reached.

2.2. Principle of Neural Networks Usage in Inverse Problems

An inverse problem can be formulated as searching for the input data knowing the output. Probably all inverse problems are based on models which can describe the response of a system to its input signal. All input data for a given model belongs to a class of admitted signals.

From the class of admitted input signals a set of typical signals $\mathbf{I}_t$ $(t=1..T)$ can be chosen . Each signal $\mathbf{I}_t$ can be modified by the model to receive the output signal $\mathbf{O}_t$. In this way pairs of $\mathbf{I}_t$, $\mathbf{O}_t$ are set up. These pairs are used for training the neural network.

In the process of training, the output signal is used as the input to the input layer of the processing elements (i.e. $\mathbf{X}=\mathbf{O}_t$). In the process of training the input signal $\mathbf{I}_t$ is

Transactions on Information and Communications Technologies vol 2, © 1993 WIT Press, www.witpress.com, ISSN 1743-3517

taken as the desired signal of the output processing elements.

A well trained net can respond to a vector **X** by proper **Y**. The vector **X** need not, naturally, come from the training data set.

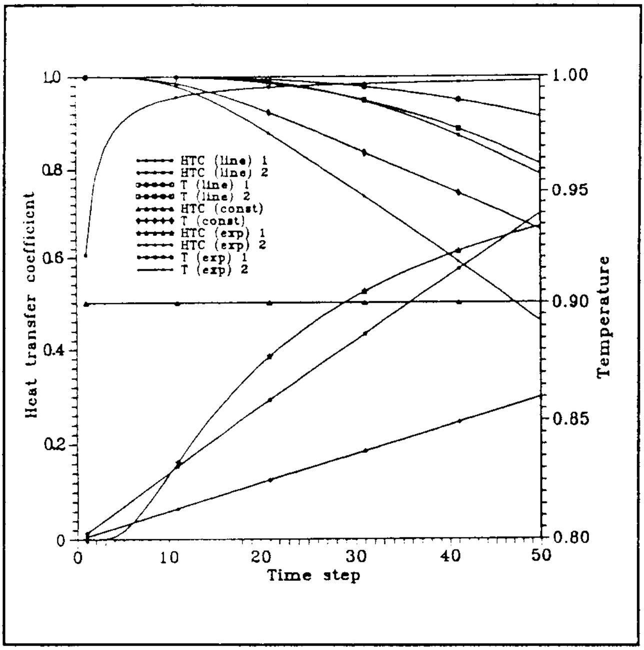

Fig.3 Input and output data to the model, T - temperature, HTC - heat transfer coefficient

2.3 Example - Inverse Heat Conduction - Function Specification

The surface $x^+ = 0$ of a one-dimensional body is subjected to time dependent boundary conditions. Boundary conditions are described in this case by the constant ambient temperature $t^+ = 0$ and the time dependent heat transfer coefficient (HTC). The starting temperature is constant and equal to 1 (dimensionless). The surface in $x^+ = 1$ is adiabatic.

Various types of heat transfer histories were tested: constant, linearly decreased and increased, step functions, exponential functions etc.

The observed time interval was divided into 50 time steps. The length of the time steps does not play a role in the described approach to the problem.

For a given type of heat transfer coefficient history, the training data was formed by a set of ten pairs of input and output vectors. The input vector contains 50 values of computed temperatures at the point $x^+ = 0.5$ and the output vector contains 50 values of appropriate heat transfer coefficients. Typical pairs of input/output data are plotted in Fig.3. It should be noted that the data are normalized to interval 0..1 in the process computation.

The net used in the tests had an input layer with 50 elements, one hidden layer with 50 elements and an output layer with 50 elements.

The results for the network trained with 3x10 pairs of input/output data (HTC: constant, linear, exponential) are shown in Fig.4. The prescribed training accuracy was

Transactions on Information and Communications Technologies vol 2, © 1993 WIT Press, www.witpress.com, ISSN 1743-3517

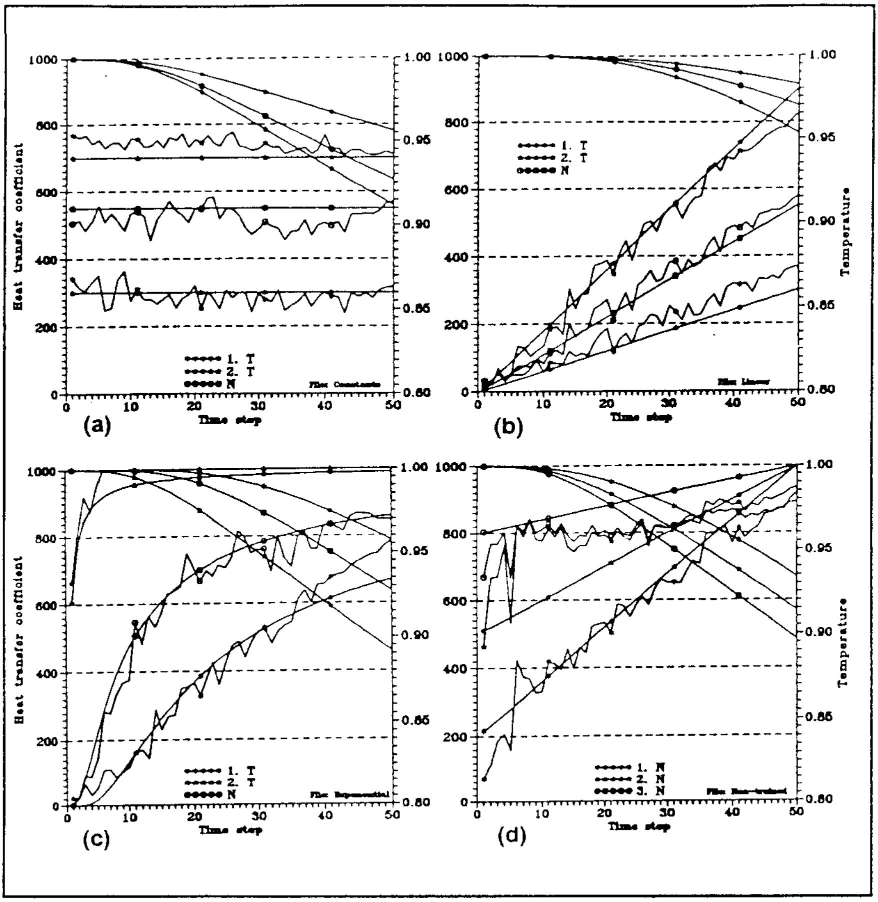

Fig.4 The response of the neural network to the input temperature history

10 %. Each of these four figures shows the input temperature histories, the exact heat transfer coefficient history and the network's response to the input data.

Figures 4a, b, c show the response to the type of data which was used for training. The curves marked **T** belong to training data. The curve **N** stands for intermediate data - that were not used for training.

Fig. 4d reports the case when the trained network is used for input data slightly different from the trained patterns. The net response becomes worse as the character of the input data differs from the character of the training data.

Transactions on Information and Communications Technologies vol 2, © 1993 WIT Press, www.witpress.com, ISSN 1743-3517

Conclusions of Neural Network Conception

Advantage	Disadvantage
□ Usage for a very wide class of inverse problems □ Short computation time □ Stability □ Suitable for n-dimensional cases □ Independence on the length of the time step (in inverse heat conduction)	□ Only restricted accuracy □ An idea of results must be known in advance □ Needs computational time in preparation phase for training

3. Expert Systems

People use natural language in order to give descriptions of the behaviour of complex systems. But it is necessary to create mathematical models for technical areas where it is impossible to use linquistic expressions. When the creation of mathematical models is impossible or expedient we can use expert systems which enable us to use linquistic expressions.

In technical practice a linguistic description of the investigated phenomenon is often used. This is true especially in cases where it is not possible or there is no reason to devise a mathematical model. Introducing L.A.Zadah's term linguistic variable permits even the processing of the above mentioned linguistic models by mathematical methods. The linguistic description of the following function is concerned:

$$y = f(x_1, x_2, \ldots, x_n).$$

This function is given by a table of linguistic values. The linguistic variables that we will work with may be considered as a fuzzy set with a real universum as we can quantify these variables (temperature, heat transfer coefficient).

In order to use the expert system we will reduce the linguistic values to triangles where just a single value exists at which the given variable gains with high certainty the given linguistic value.

A sentence that will be named a rule is composed of the names of the linguistic variables, of their linguistic values and logical conjunctions. Joining the rules with the help of the logical conjunctions gives a linguistic model.

The software system LMPS (Linquistic Model Processing System) was used for the solution of the inverse heat transfer conduction problem. The system was developed by the firm JANES (Brno, Czech Republic). The main advantage of this system is that is uses very simple but highly effective mathematical methods (see below). The application of fuzzy sets, multi-valued logic, linquistic model compilation and rule formulation are described in [4].

3.1 Mathematical base of LMPS

The rule based expert system LMPS was used for inverse task solving. This expert system uses rules of the IF-THEN type with variable semantic interpretation in Lukaszewics many valued logic [4].

A knowledge base of the following form (2) was created, where x_i, y are linguistic variables and L_{ji}, Kj their linguistic values. Every linguistic value is supposed to be a convex normalized fuzzy set with a real universum.

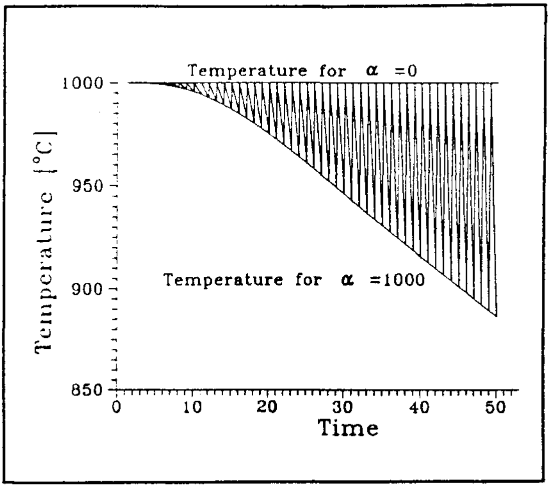

Fig.5 The working area for solutions

$$\begin{aligned}&\text{IF } x_1 = L_{11},\ x_2 = L_{12}, \dots x_n = L_{1n} \text{ THEN } y = K_1 \\ &\text{IF } x_1 = L_{21},\ x_2 = L_{22}, \dots x_n = L_{2n} \text{ THEN } y = K_2 \\ &\dots \\ &\text{IF } x_1 = L_{m1},\ x_2 = L_{m2}, \dots x_n = L_{mn} \text{ THEN } y = K_m\end{aligned} \quad (2)$$

The following forms of semantic interpretation of the knowledge base (2) were used:

1) CIC-model (contradiction sensitive, redundance insensitive)

$$\begin{aligned}&((\mu_{11} \text{ and } \mu_{12} \text{ and } \dots \text{ and } \mu_{1n}) => \mu_1) \text{ and} \\ &((\mu_{21} \text{ and } \mu_{22} \text{ and } \dots \text{ and } \mu_{2n}) => \mu_2) \text{ and} \\ &\dots \\ &((\mu_{m1} \text{ and } \mu_{m2} \text{ and } \dots \text{ and } \mu_{mn}) => \mu_m)\end{aligned}$$

2) CI&-model (contradiction sensitive, redundance sensitive)

$$\begin{aligned}&((\mu_{11} \text{ and } \mu_{12} \text{ and } \dots \text{ and } \mu_{1n}) => \mu_1)\ \& \\ &((\mu_{21} \text{ and } \mu_{22} \text{ and } \dots \text{ and } \mu_{2n}) => \mu_2)\ \& \\ &\dots \\ &((\mu_{m1} \text{ and } \mu_{m2} \text{ and } \dots \text{ and } \mu_{mn}) => \mu_m)\end{aligned}$$

3) CCD-model (contradiction and redundance insensitive)

$$\begin{aligned}&((\mu_{11} \text{ and } \mu_{12} \text{ and } \dots \text{ and } \mu_{1n}) \text{ and } \mu_1) \text{ or} \\ &((\mu_{21} \text{ and } \mu_{22} \text{ and } \dots \text{ and } \mu_{2n}) \text{ and } \mu_2) \text{ or} \\ &\dots \\ &((\mu_{m1} \text{ and } \mu_{m2} \text{ and } \dots \text{ and } \mu_{mn}) \text{ and } \mu_m)\end{aligned}$$

3.2 Principle of Expert System Usage in Inverse Problem

Before questioning the expert system, it is necessary to fill its knowledge base with some quantity of high-quality information. The knowledge base is a data file holding linquistic variable definitions, linquistic values, rule declarations, author information and further auxilliary information. The knowledge base was filled with rules generated from data calculated from the cooling produced by diffrent values of the heat transfer coefficient. This calculation gave coresponding heat transfer coefficients [α] and temperature course couples.

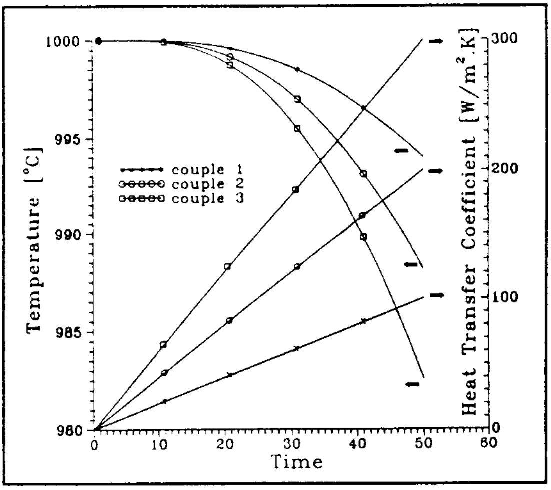

Fig.6 The temperature history before transformation

The minimum value of α was taken to be 0 [W.m⁻².K⁻¹] and the maximum 1000 [W.m⁻².K⁻¹]. These boundary values together with coresponding temperature courses define the working area for all the solutions. This is shown in Fig.5.

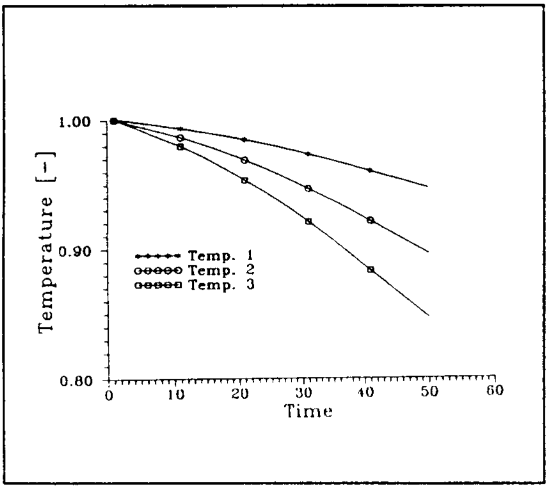

Fig.7 The temperature history after transformation

The temperatures were transformed into the interval [0,1] before fuzzification since there are very few differences at the start of the temperature course. The temperature course before and after this transformation is shown in Fig.6 and Fig.7.

Time was not used as an independent variable, but the values of α and temperature were bound to certain points whose position in the course carried information about time. Only values of temperature prior to the point for which α must be found are seen in the rules. This is because values of temperature after this point cannot (for physical reasons) influence the value of α. The fuzzified values are shown in Fig.8. In the paragraph 3.3 is shown the arrangement of the rule.

When we have filled knowledge base we can question the system in order to find a solution. To ask a question, we set a number of variables to known values. In our case, these variables represent a temperature course. From these variables we find the value of α. We must repeat this question for every point of α's history, because LMPS is only capable of answering a question at one point. The values of α at other points is not exploited by the system.

The answer of the system is a fuzzy set in a graphical form, as shown in Fig.9. Because we need to obtain only a single value of α, we must perform defuzzification. The defuzzification procedure we use is the 'centre of mass' method.

3.3 The arrangement of the rule

The ilustration of the rule for one value of α is shown below.

if(temperature0 is T10 and
temperature1 is T10 and
temperature2 is T9 and
...
temperature11 is T4 and
temperature12 is T3) then koefficient α12 is A8

The same rule must be compiled for all points, at which we watch the temperature history. We choose the temperature histories which perfectly cover the whole working space.

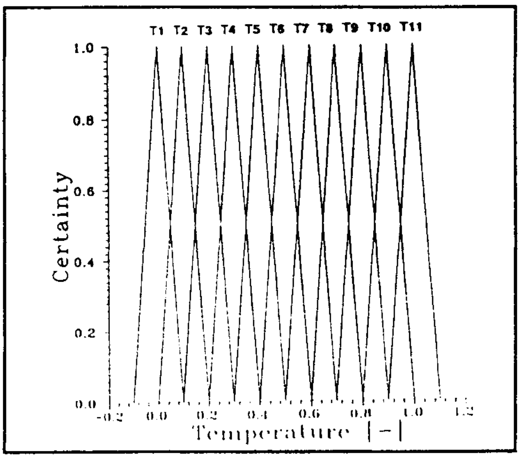

Fig.8 The fuzzified value

4. Conclusion

A common conclusion can be made based on results from the usage of neural networks and expert systems:

The neural network gives a response even if the net is not trained for the input data. The system is nonlinear and does not yield an error message when a given question has not been defined in the training data.

The expert system is able to give very precise information if the working space is well covered by the rules. The system informs the user that the answer has high or low certainty. This information can be used for suplementation of the knowledge base.

This paper shows that this new approach to inverse problems gives reasonable results. This methods can replace the clasical inverse methods.

Transactions on Information and Communications Technologies vol 2, © 1993 WIT Press, www.witpress.com, ISSN 1743-3517

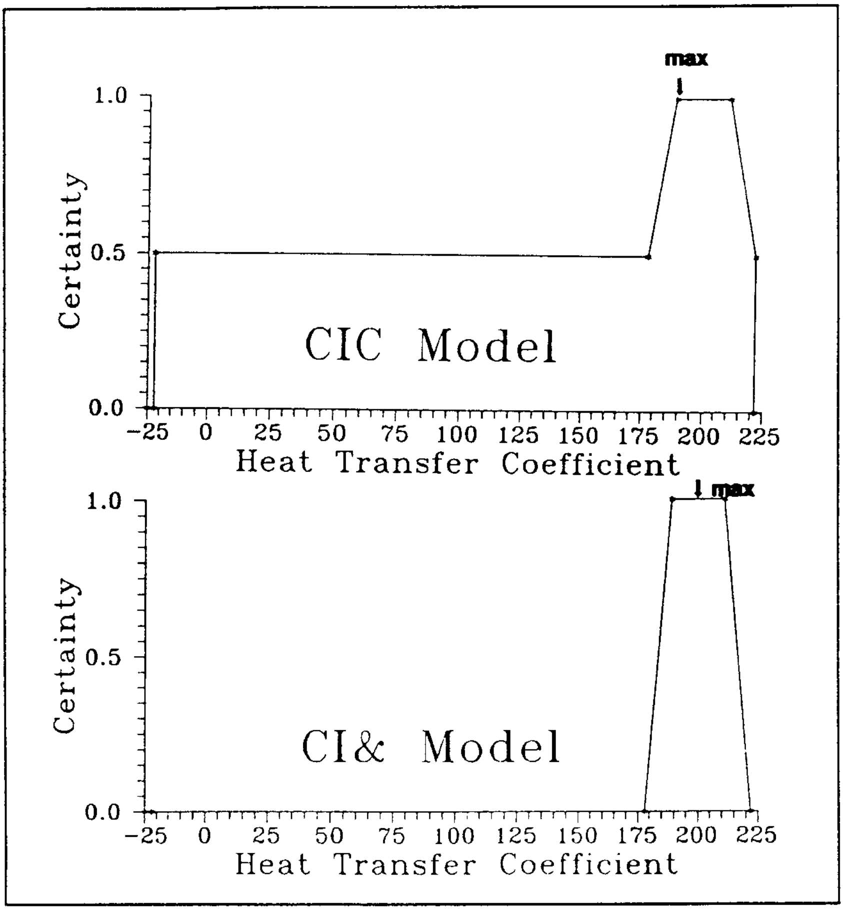

Fig.9 The answer of the expert system

5. References

[1] J. V. Beck, B. Blackwell, and C. St. Clair,"Inverse Heat Conduction: Ill-posed Problems". New York: Wiley-Interscience, 1985.

[2] M. Raudenský, V. Dumek: "Inverse Task for Determining Heat Transfer Coefficient, Strojnický časopis, No. 1, 1992, (Journal of Czechoslovak Academy of Science).

[3] M.Druckmüller, Many-valued logic system for linquistic model processing. Int.J.of General Systems, 1991, 20, 1, 31-38.

[4] LMPS - Reference Guide, Brno, 1992