Advanced Polynomial Neural Networks Architecture with New Adaptive Nodes

Sung-Kwun Oh, Dong-Won Kim, Byoung-Jun Park, and Hyung-Soo Hwang

Abstract: In this paper, we propose the design procedure of advanced Polynomial Neural Networks(PNN) architecture for optimal model identification of complex and nonlinear system. The proposed PNN architecture is presented as the generic and advanced type. The essence of the design procedure dwells on the Group Method of Data Handling (GMDH). PNN is a flexible neural architecture whose structure is developed through learning. In particular, the number of layers of the PNN is not fixed in advance but is generated in a dynamic way. In this sense, PNN is a self-organizing network. With the aid of three representative numerical examples, comparisons show that the proposed advanced PNN algorithm can produce the model with higher accuracy than previous other works. And performance index related to approximation and generalization capabilities of model is evaluated and also discussed.

Keywords: polynomial neural networks(PNN), group method of data handling (GMDH), self-organizing network, approximation and generalization capabilities

I. Introduction

The mathematical models to express dynamic analysis of nonlinear real system, have had lots of difficulties in the selection of variables constructing the model among many input-output variables. Moreover, high-order equation requires a large amount of data for estimating all system parameters in mathematical models. So it needs the model designer who has had the specific and prior knowledge of the model architecture or its components. In that case, it is impossible to make the high performance model architecture or to perform process control very well when we depend on the specific and prior knowledge of designer and experiment too much. To deal with such a problem, GMDH[1] was developed in Russia in the late 1960's by Ivakhnenko as an analysis technique for identifying nonlinear relations between system inputs and outputs. The primary disadvantage of GMDH is that it not only can generate a complex polynomial even for some simple system but also can not take into consideration of input-output relationship well because of its limited architecture, and if there is a sufficiently large number of training data, GMDH has a tendency to produce overly complex networks as it tries to stretch for the last bit of accuracy. So it can be shown that the GMDH is very ineffective in modeling nonlinear systems having different characteristics in different environments.

In order to overcome the limitations of GMDH, we propose a PNN algorithm and show its design procedure for optimal PNN model. The number of input variables used in Partial Description(PD) of each node is extended and the order of regression polynomial used in PD is also extended as linear, quadratic, and cubic. And also the number of input variables and the order of regression polynomial are not fixed and can be changed in each layer of PNN. Through the extended regression polynomial, the architecture of PNN can be changed to adapt to system environment. Two types of architectures, the basic PNN and modified PNN architecture, are studied in this paper.

II. PNN algorithm

PNN algorithm that is based on GMDH method can utilize other mathematical forms such as linear, modified quadratic, cubic, high-order polynomial and so on.

By choosing the really significant input variables and polynomial order among these various kinds of forms, we can obtain the best ones from the extracted partial descriptions according to both selecting nodes of each layer and generating additional layers until the best performance is taken. A new methodology which includes these design procedure leads us to get the optimal PNN architecture for identification of nonlinear system. The input-output data are given as follows.

$$(\mathbf{X}_i,\ y_i) = (x_{1i},\ x_{2i},\ \ldots,\ x_{Ni},\ y_i), \qquad i = 1,\ 2,\ 3,\ \ldots,\ n\,.$$

The input-output relationship of the above data by PNN algorithm can be described in the following way

$$y = f(x_1,\ x_2,\ \ldots,\ x_N)\,. \tag{1}$$

The estimated output $\hat{y}$ of the above output y is as follows

$$\hat{y} = \hat{f}(x_1, x_2, \cdots, x_N) = c_0 + \sum_{k1} c_{k1} x_{k1} + \sum_{k1k2} c_{k1k2} x_{k1} x_{k2} + \sum_{k1k2k3} c_{k1k2k3} x_{k1} x_{k2} x_{k3} + \ldots \tag{2}$$

Where, c_k denotes coefficients.

To get the estimated output $\hat{y}$, we construct a PD for each pair of independent variables in first generation according to number of input variables. We determine coefficients of PD by the least squares method using training data set and then the choice of optimal estimated model is implemented in the first layer and compute a PD from new intermediate variables(for example z_m)generated from the next generation. After that, we take another pair of new input variables, and repeat operation until the stop condition of PNN algorithm is satisfied in each generation. PNN algorithm can be described as the procedure of eight steps. Each step is as follows:

Manuscript received: Aug. 12, 2000., Accepted: Jan. 3, 2001.

Sung-Kwun Oh, Dong-Won Kim, Byoung-Jun Park, Hyung-Soo Hwang: School of Eelectrical, Electronics and Information Engineering, Wonkwang University(ohsk@wonnms.wonkwang.ac.kr/ upground@gaebyok.wonkwang.ac.kr/lcap@guebyok.wonkwang.ac.kr/ hshwang@wonnms.wonkwang.ac.kr)

※ This paper was supported by Wonkwang University in 1999.

Step 1: Determine system input variables

Define the input variables as x_i, i=1 ,2, ⋯ , n related to output variable y. If needed, the normalization of input data is done. Depending on the no. of system input variables as shown below, two types of PNN architectures are considered for performance improvement of PNN model.

Refer Figs. 1-2.

a) In case that the no. of system input variables is 2 or 3, the advanced type is used

b) In case that the no. of system input variables is 3 or more, the generic type is used.

Step 2: Division of input and output data set

The input and output data sets $(\mathbf{X}_i, y_i) = (x_{1i}, x_{2i}, \ldots, x_{Ni}, y_i)$, $i = 1,2,3,\ldots,n$ are divided into two parts of the training data subset n_{tr} and the testing data subset n_{te}. The training data subset is used for obtaining PNN model by estimating the coefficients of the PD of nodes in each layer of PNN. The testing data subset is used to evaluate the PNN model estimated using the training data subset, and to construct the PNN model with better prediction ability for ranking and selecting the PDs of that estimated model of each layer from the viewpoint of mean squared error. Where, $n = n_{tr} + n_{te}$

Step 3: Decision of PNN architecture

PNN architecture is decided according to the number of input variables and the order of PD in each layer. Two kinds of PNN architectures, which are the basic PNN and the modified PNN architecture, are presented and also two cases for each architecture are used.

Accordingly, with respect to the PNN architecture, we consider as follows.

a) Basic PNN architecture – The number of input variables of PDs is same in every layer.

Case 1: The polynomial order of PDs is same in every layer.

Case 2: The polynomial order of PDs in 2nd layer or more has a different or modified type in comparison with that one of PDs in 1st layer.

b) Modified PNN architecture – The number of input variables of PDs is different in each layer.

Case 1: The polynomial order of PDs is same in every layer.

Case 2: The polynomial order of PDs in 2nd layer or more has a different or modified type in comparison with that one of PDs in 1st layer.

The specific feature of the modified PNN architecture is that not only the order but also the number of independent input variables does not remain same in PDs of each layer but is expanded to various kinds of types in the next new layer. Therefore the complex PDs as well as the simple PDs can be utilized effectively according to various kinds of modified PNN architectures by taking into consideration of both compactness and mutual input-output relationship of each layer.

Two types, a) the generic and b) the advanced type, of the basic and modified PNN architectures are shown in Figs. 1-2, where **z'**$_i$(Case 2) of the 2nd layer denote that polynomial order of the PD of each node has a different or modified type each other in comparison with **z**$_i$(Case 1) of the 1st layer.

In b) the advanced type of Figs. 1-2, the node of dotted line means the nodes of the previous layer. The superscript A of $\mathrm{PD_B}^{\mathrm{A}}$ denotes the layer number and the subscript B of $\mathrm{PD_B}^{\mathrm{A}}$ denotes the node sequence number of new nodes generated by the combination of the node outputs(outputs of PDs) of the preceding layer at the Ath layer.

Step 4: Determine the no. of input variables and order of a PD

We determine the regression polynomial architecture of a PD related to PNN architecture and refer to Table 2. We choose the input variables of a node from N input variables $x_1, x_2, \ldots, x_N$. The total number of PDs of current layer depends on the number of the selected input variables from nodes of preceding layer. This results in $k = N!/(N-r)!r!$ nodes, where N is the total number of the independent variables and r is the number of the chosen input variables. The choice of both the input variables and the order of a PD helps to select the fittest estimated model with regard to characteristics of system, model design strategy, nonlinearity and predictive capability as shown in Table 1.

Table 1. PNN architecture.

PD Type / Layer	No. of input variables	Order of Polynomial	PNN architecture

1st layer	p	P	p=q: **Basic PNN**
			P=Q: Case 1
			P≠Q: Case 2
			p≠q: **Modified PNN**
			P=Q: Case 1
			P≠Q: Case 2

2-5th layer	q	Q	

(p=2, 3, 4, q=2, 3, 4 ; P=1, 2, 3, Q=1, 2, 3)

Step 5: Estimate the coefficients of a PD

The coefficient vector $\mathbf{C}_i$ is obtained by solving (5) to minimize the mean squared error between y_i and z_{mi} of eq. (3)

$$E = \frac{1}{N_{tr}} \sum_{i=0}^{N_{tr}} (y_i - z_{mi})^2 . \tag{3}$$

Using the training data subset, n_{tr}, the set of output equations can be represented in each layer

$$\mathbf{Y} = \mathbf{X}_i \mathbf{C}_i . \tag{4}$$

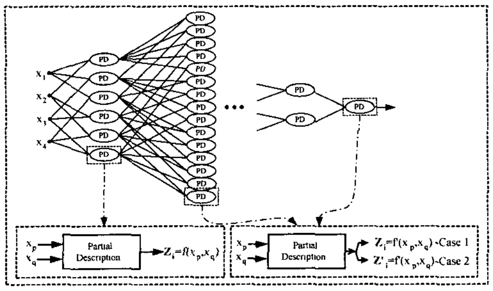
The coefficients of PD of nodes in each layer are determined by the standard least squared method as follows

$$\mathbf{C}_i = (\mathbf{X}_i^T \mathbf{X}_i)^{-1} \mathbf{X}_i^T \mathbf{Y} . \tag{5}$$

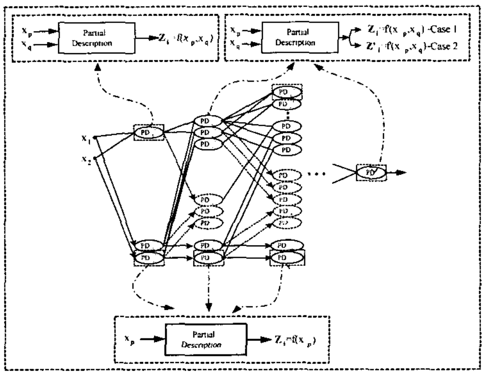
Where

$$\mathbf{Y} = [y_1 \, y_2 \cdots y_{n_{tr}}]^T, \mathbf{X}_i = [X_{1i} \, X_{2i} \cdots X_{ki} \cdots X_{n_{tr}i}]^T,$$
$$\mathbf{X}^T{}_{ki} = [1 \, x_{ki1} \, x_{ki2} \cdots x_{kin} \cdots x_{ki1}^m \, x_{ki2}^m \cdots x_{kin}^m]$$
$$\mathbf{C}_i = [c_{0i} \, c_{1i} \, c_{2i}, \ldots, c_{n'i}]^T$$

i : Node number, n_{tr} : Number of the training data subset, n : Number of the selected input variables, m : maximum order, n' : Number of estimated coefficients except for constant term.



(a) Generic type



(b) Advanced type

Fig. 1. Configuration of the basic PNN architecture (The generic and advanced PNN).

This procedure is implemented repeatedly for all nodes of the layer and also for all layers of PNN from input layer to output layer.

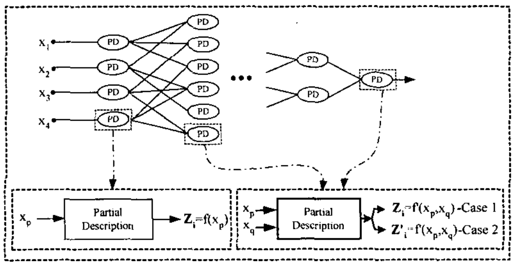
Step 6: Choosing PDs with better predictive capability

We determine the total number, $N!/(N-r)!r!$ of PDs according to combinations of nodes in each layer. Each PD which was estimated using the training data subset is evaluated by computing identification error of the mean squared error using the testing data subset. And we compare those evaluated output and choose several PDs which have better output performance. Here, we use the pre-defined number W of PDs with better predictive capability that must be preserved for optimal operation of the next generation in the PNN algorithm. The outputs of the preserved PDs(called Survivors), serve as inputs to the next layer(generation).

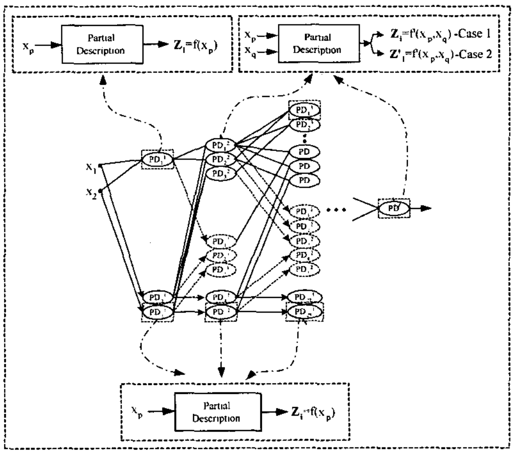
There are two cases for the decision of the number of the preserved PDs in each layer

a) If $N!/(N-r)!r! < W$, then for the next layer, the number of the preserved PDs is $N!/(N-r)!r!$

b) If $N!/(N-r)!r! \geq W$, then for the next layer, the number of the preserved PDs is W



(a) Generic type



(b) Advanced type

Fig. 2. Configuration of the modified PNN architecture (The generic and advanced PNN).

Step 7: Stop condition

Two methods for stop condition of PNN algorithm are commonly available.

a) Method 1: The stop condition shown in eq. (6) indicates that an optimal PNN model was reached at the previous layer, and the modeling process is stopped. The stop condition of method 1 can be described as follows:

$$E_j \geq E_* \qquad (6)$$

Where E_j ; minimal identification error of the current layer, E_* ; minimal identification error of previous layer

b) Method 2: PNN algorithm terminates when the number of generation predetermined by the designer is reached. If the procedure for PNN modeling is stopped according to the Method 1, PNN models with better performance are ignored under generation of layers. And if there happens a sufficiently large number of input variables, PNN architecture has a tendency to produce overly complex networks. So it would take too much time and it would require too much computer memory. To compensate for these difficulties, we take into consideration of both a stop condition for better performance and the number of generation predetermined by designer. The network generation of PNN architecture is stopped at most in the predetermined layer for simple and efficient modeling. This method helps to achieve a balance between model accuracy and complexity.

Step 8: Setting new input variables in the next layer

If E_j (the minimum value in the current layer) is not satisfied for a stop condition, it indicates that progress towards the best model is still being continued. The outputs of the preserved PDs serve as new inputs to the next layer. It can be described as the following

$$x_{1i} = z_{1i}, x_{2i} = z_{2i}, \ldots, x_{wi} = z_{wi} . \qquad (7)$$

And PNN algorithm is carried out repeatedly from step 4 to 8 for generation of new estimated outputs z's as inputs to the next layer.

III. Simulation and discussion of results

According to each step shown in the preceding section, the PNN method is illustrated with the aid of the well-known problem of the nonlinear system proposed in [6], sewage treatment process[8][11] and gas furnace process[15].

1. Nonlinear static system

In this section, we perform a simulation to illustrate the validity of the proposed algorithm.

The training data in this example are obtained from a two-input nonlinear equation defined by

$$y = (1 + x_1^{-2} + x_2^{-1.5})^2,\ 1 \le x_1, x_2 \le 5\,. \qquad (8)$$

This nonlinear static equation is widely used to evaluate modeling performance. This equation was also used by Sugeno and Yasukawa [6], Nakanishi [7], and Kim[12][13] to test their modeling approaches.

This system represents the nonlinear characteristic as shown in Fig. 3, which shows a three-dimensional input-output graph of this system. From this system equation, 50 input-output data are obtained.

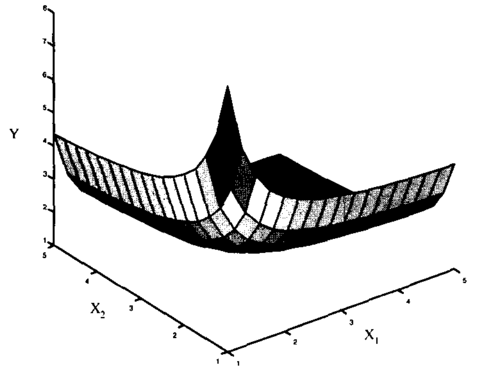


Fig. 3. Input-output relation of nonlinear system.

To allow comparison, we use the same performance index adopted in refs. [6]:

$$PI = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2\,. \qquad (9)$$

Where m is the number of data pairs, and y_i and $\hat{y}_i$ are the i the desired output and model output, respectively.

In this simulation, the proposed polynomial neural networks architecture is used to build the optimal model. Because only two system input variables are considered, it is difficult to generate the polynomial neural networks of this nonlinear system. Therefore, the PNN architecture uses the advanced Type shown in Fig. 4.

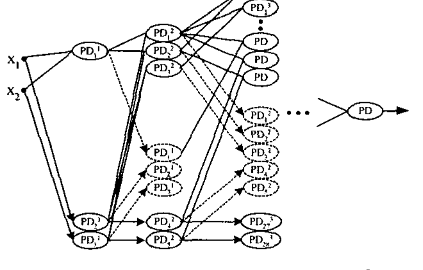


Fig. 4. PNN architecture with 2 system inputs for nonlinear system's identification.

Table 2. Form of regression polynomials used in PNN architecture.

No. of input Variables	Order of Polynomial	Node equations considered
2	Type 1	$c_0+c_1x_1+c_2x_2$
	Type 2	$c_0+c_1x_1+c_2x_2+c_3x_1^2+c_4x_2^2+c_5x_1x_2$
	Type 3	$c_0+c_1x_1+c_2x_2+c_3x_1\,x_2$
3	Type 1	$c_0+c_1x_1+c_2x_2+c_3x_3$
	Type 2	$c_0+c_1x_1+c_2x_2+c_3x_3+c_4x_1^2+c_5x_2^2$ $+c_6x_3^2+c_7x_1x_2+c_8x_1x_3+c_9x_2x_3$
	Type 3	$c_0+c_1x_1+c_2x_2+c_3x_3+$ $c_4x_1x_2+c_5x_1x_3+c_6x_2x_3$

▶ The Basic PNN architecture

Case 1: This case is that the number of input variables of PDs is same in every layer and the polynomial order of PDs is also same in every layer.

The training result plotted in Figs. 5 depend on both the number of node input variables and three different types of polynomial order such as linear, quadratic, or modified quadratic, that is, Type 1, Type 2, or Type 3. Here Type 1, Type 2, and Type 3 are listed in Table 2. Fig. 5 shows the performance index (identification error) of PNN architecture with 2 node inputs and Type 1, Type 2, and Type 3. In that case the best result of 2 node inputs, PI=0.0212 is obtained by using Type 2.

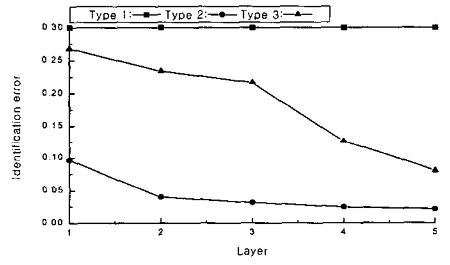


Fig. 5. Identification error (Every layer: 2 inputs).

Case 2: This case is that the number of input variables of each PD is same in every layer but the polynomial order of PDs in the 2nd layer or more has a different or modified type in comparison with that one of PDs in the 1st layer.

The training result is plotted in Fig. 6. In Fig, 'Type a → b'(a, b=1, 2, 3) means that the polynomial order of PDs

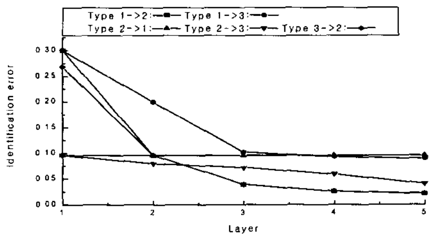


Fig. 6. Identification error (Every layer: 2 input).

changes from Type a in the 1st layer to Type b in the 2ND layer or more.

Fig. 6 show the performance index of PNN architecture with 2 node inputs and Type 1→2, Type 1→3, Type 2→1, Type 2→3, and Type 3→2. In that case the best result of 2 node inputs, PI=0.0212 is obtained by using Type 1→2 .

► The Modified PNN architecture

Case 1: This case is that the number of input variables of each PD in the 1st layer is different in comparison with that one in the 2nd layer or more and the polynomial order of each PD is same in every layer.

Fig. 7 shows the identification error of modified PNN in Case 1 when the number of input variables is 2 in 1st layer and that of input variables in 2nd layer or more is 3 with Type 1, Type 2, and Type 3. The best result, PI=0.00041, is achieved by using Type 2.

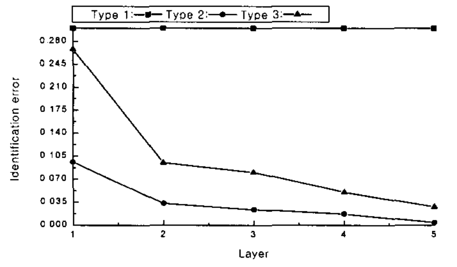


Fig. 7. Identification error(1st layer: 2 input, 2nd layer or more: 3 inputs).

Case 2: This case is that the number of input variables of each PD in the 1st layer is different in comparison with that one in the 2nd layer or more. And also the polynomial order of each PD in the 2nd layer or more has different or modified type in comparison with that one in the 1st layer.

Fig. 8 shows the identification error of modified PNN in Case 2 when the number of input variables is 2 in 1st layer and that of input variables in 2nd layer or more is 3 with Type 1→2, Type 1→3, Type 2→1, Type 2→3, and Type 3→2. The best result, PI=0.0105, is achieved by using Type 3→2.

Table 3. shows a comparison of identification errors with previous modeling methods. The experiment results show output performances of the proposed PNN model according to two kinds of PNN architectures.

As we know from Table 3, the performance results of PNN architecture are quite satisfactory. Compared with approaches

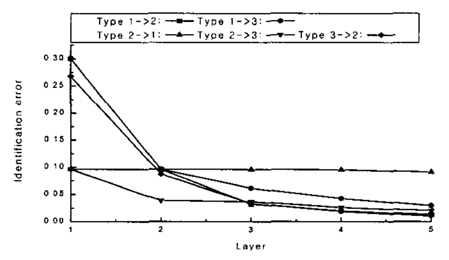


Fig. 8. Identification error(1st layer: 2 input, 2nd layer or more: 3 inputs).

presented in previous literatures [6][12]-[14], our modeling method has much more accuracy.

Table 3. Comparison of identification error with previous modeling methods.

Model			MSE
			PI
Sugeno and Yasukawa [6]			0.079
Kim et al. [12]			0.019
Kim et al. [13]			0.0089
Gomez-Skarmeta et al. [14]			0.070
Our model	Basic PNN	Case 1	0.0212
		Case 2	0.0212
	Modified PNN	Case 1	0.0041
		Case 2	0.0105

2. Sewage treatment process.

Sewage treatment generally uses the activated sludge process which consisted of a sand basin, primary sedimentation basin, aeration tank and final sedimentation basin(see Fig.9).

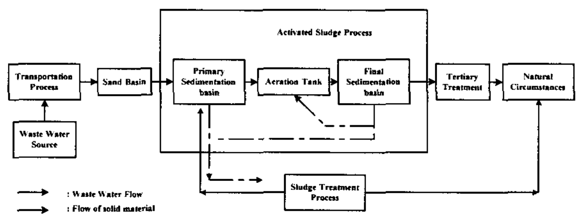


Fig. 9. Configuration of the sewage treatment system.

The suspended solid included in the sewage is sedimented by gravity in sand and primary sedimentation basins. Air is consecutively absorbed in the sewage in the aeration tank for several hours. Microbe lump(that is called floc or activated sludge) springing naturally, mainly removes the organic matter from the aeration tank. Activated sludge biochemically oxygenates, proliferates and resolves the organic matters into hydrogen and carbon dioxide by metabolism. In the final sedimentation basin, floc is sedimented, recycled and again used to remove the organic matter and then purified water is transported to the tertiary sedimentation basin.

The activated sludge process is the process that involves an aeration tank and final sedimentation. We measure the biological oxygen demand(bod) and the concentration of suspended solid(ss) in influent sewage at the primary sedimentation basin, and effluent bod(ebod)and ss(ess) in the effluent sewage at the final sedimentation basin. Because ebod and ess are changed, depending on the bod and ss, dissolved oxygen set-point(dosp) and recycle sludge ratio setpoint(rrsp) are set so that ess and ebod should be kept up less than the prescribed small quantity. Ebod and ess depend on mixed liquid suspended solid(mlss), waste sludge ratio(wsr), rrsp and dosp. Bod has a correlation with ss.

In this paper, a sewage treatment system plant in Seoul, Korea, is chosen as a model. The Self-organizing Polynomial Neural Networks modeling by two kinds of architecture, the basic PNN and the modified PNN, is carried out using 52 pairs of input-output data obtained from the activated sludge process. The performance index used in the computer simulation will be the same as eq. (9). The experiments were completed for four fundamental architectures of the PNNs and the results are shown in a series of figures.

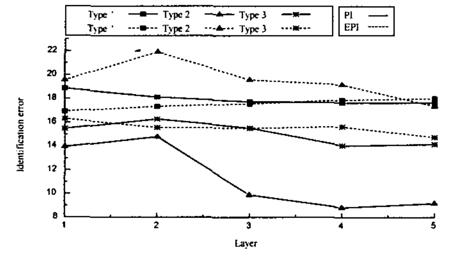


Fig. 10. Identification error of basic PNN in Case 1 (Every layer: 2 inputs).

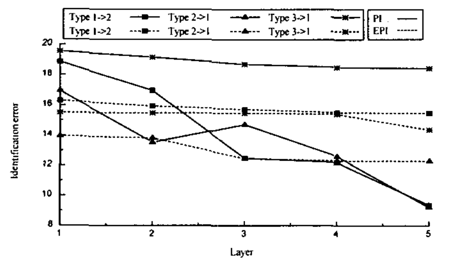


Fig. 11. Identification error of basic PNN in Case 2 (Every layer: 2 inputs).

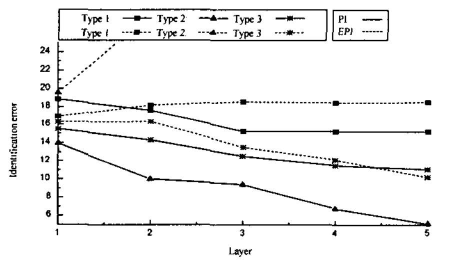


Fig. 12. Identification error of modified PNN in Case 1 (1st layer: 2 inputs, 2nd layer or more: 3 inputs).

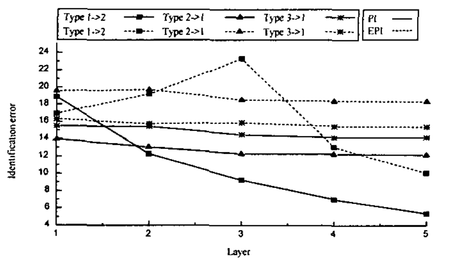


Fig. 13. Identification error of Modified PNN in Case 2 (1st layer: 2 inputs, 2nd layer or more: 3 inputs).

Table 4 summarizes the results of comparative analysis of PNNs and other models. Again, the performance of the PNN is far better both in the sense of its prediction (generalization) and approximation abilities.

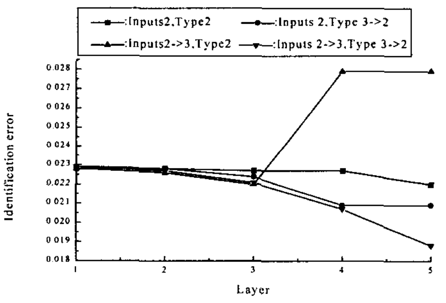
Table 4. Comparison of identification error with previous modeling methods.

Model			MSE	
			PI	EPI
Oh and Pedrycz's model [8]	Simplified		12.802	15.915
	Linear		6.396	54.233
Oh's model [11]	FNN-(GA)		10.163	14.698
	FNN-(GA+HCM)		9.920	10.129
Our Model	Basic PNN	Case 1	9.158	17.367
		Case 2	9.353	9.211
	Modified PNN	Case 1	11.053	10.192
		Case 2	5.388	10.083

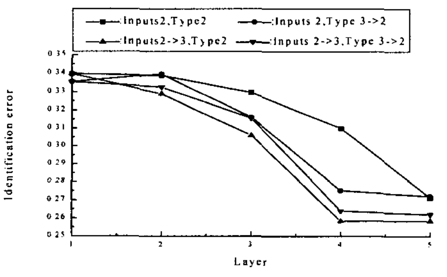
3. Gas furnace process

The proposed method is applied to the time-series data of a gas furnace utilized by Box and Jenkins[15]. The delayed terms of methane gas flow rate, u(t-3) and carbon dioxide density, y(t-1) are used as system input variables. And as output variable, y(t) is used. We choose the input variables of nodes in the 1st layer of PNN architecture from these system input variables. The total data set consisting of 296 input-output pairs was split into two parts. The first one (consisting of 148 pairs) was used for training. The remaining part of the series serves as a testing set. We consider the MSE (eq. 9) being regarded here as a performance index.

Again, a series of comprehensive experiments was conducted for all four main architectures of the PNNs with the results summarized in the following Figure.



(a)



(b)

Fig. 14. Identification error of PNNs. (a) PI, (b) EPI

Table 5 contrasts the performance of the PNN network with other fuzzy models studied in the literature. The experimental results clearly reveal that the PNN outperforms the existing models both in terms of better approximation capabilities (lower values of the performance index on the training data, PI_s) as well as superb generalization abilities (expressed by the performance index on the testing data EPI_s).

Table 5. Comparison of identification error with previous fuzzy models (PI- performance index over the entire data set, PI_s- performance index on the training data, EPI_s – performance index on the testing data).

Model			MSE		
			PI	PI_s	EPI_s
Box and Jenkins' model[15]			0.710		
Sugeno and Yasukawa' model[6]			0.190		
Xu's model[16]			0.328		
Pedrycz's model[17]			0.320		
Oh and Pedrycz's model[8]			0.123	0.020	0.271
Kim, et al.'s model[13]				0.034	0.244
Lin and Cunningham's model[18]				0.071	0.261
Our model	Basic PNN	Case 1		0.021	0.271
		Case 2		0.020	0.272
	Modified PNN	Case 1		0.027	0.258
		Case 2		0.018	0.262

IV. Conclusions

In this paper, the design procedure of Polynomial Neural Networks(PNN) is proposed to build an optimal model architecture for nonlinear and complex system modeling. Nonlinear static system, sewage treatment process data, and gas furnace data are used for the purpose of evaluating the performance of the proposed PNN modeling method. Experimental results show that the proposed method is superior to other previous works from the viewpoint of the identification errors.

The feature of the proposed method includes the following.

1) The conflict between overfitting for approximation and generalization can be avoided from the proposed PNN algorithm.

2) Depending on the characteristics of nonlinear system, several kinds of PNN architectures can be decided.

3) In spite of small data sets of nonlinear systems, it is possible to obtain the optimal PNN architecture to be able to adapt to system environment

4) The number of input variables and the polynomial order of PD can be chosen flexibly for the PNN modeling with higher accuracy and better generalization ability

References

[1] A. G. Ivahnenko, "Polynomial theory of complex systems", *IEEE Trans. On Systems, Man and Cybernetics*, vol. SMC-1, pp. 364-378, 1971.

[2] A. G. Ivahnenko and H. R. Madala, *Inductive Learning Algorithms for Complex Systems Modeling*, CRC Press, London, 1994.

[3] D. M Steiger and R. Sharda, "Analyzing mathematical models with inductive learning networks", *European Journal of Operational Research* 93, 387-401, 1996.

[4] Ya. Pachepsky, W. Rawls, D. Gimenez, and J. P. C. Watt, "Use of soil penetration resistance and group method of data handling to improve soil water retention estimates", *Soil &Tillage Research*, vol. 49, pp. 117-126, 1998.

[5] S. J. Farlow, *The GMDH algorithm, In: S.J. Farlow (Ed), Self-organizing Methods in Modeling: GMDH Type Algorithms*, Marcel Dekker, New York, pp. 1-24, 1984.

[6] M. Sugeno and T. Yasukawa, "A fuzzy-logic-based approach to qualitative modeling", *IEEE Trans. Fuzzy Systems*, vol. 1, no. 1, 7-31, 1993.

[7] H. Nakanishi, I. B. Turksen, and M. sugeno, "A review and comparison of six reasoning methods", *Fuzzy sets and Systems*, vol. 57, pp. 257-294, 1992.

[8] S. K. Oh and W. Pedrycz, "Identification of fuzzy systems by means of an Auto-Tuning algorithm and its application to nonlinear systems", *Fuzzy Sets and Systems*, vol. 115, no. 2, pp. 205-230, 2000.

[9] C. S. Park, S. K. Oh, and W. Pedrycz, "Fuzzy identification by means of Auto - Tuning algorithm and weighting factor", *The Third Asian Fuzzy Systems Symposium(AFSS)*, pp. 701-706, 1998.

[10] S. K. Oh, T. C. Ahn, and W. Pedrycz, "Fuzzy polynomial neural networks-based architecture and its application to nonlinear process systems", *Seventh IFSA World Congress, Prague*, pp. 495-499, 1997.

[11] S. K. Oh, K. C. Yoon, and H. K. Kim, "The design of optimal Fuzzy-Neural networks architecture by means of GA and an aggregate weighted performance index", *Journal of Control, Automation and Systems Engineering*, vol. 6, no. 3, March, 2000.

[12] E. T. Kim, M. K. Park, S. H. Ji, and M. Park, "A new approach to fuzzy modeling", *IEEE Trans. Fuzzy Systems*, vol. 5, no. 3, pp. 328-337, 1997.

[13] E. Kim, H. Lee, M. Park, and M. Park, "A simple identified Sugeno-type fuzzy model via double clustering", *Information Sciences* 110, 25-39, 1998.

[14] A. F. Gomez-Skarmeta, M. Delgado, and M. A. Vila, "About the use of fuzzy clustering techniques for fuzzy model identification", *Fuzzy Sets and Systems*, vol. 106, 179-188, 1999.

[15] G. E. P. Box and F. M. Jenkins, "Time series analysis : forecasting and control", 2nd ed. Holden-day, 1976.

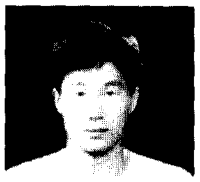
[16] C.W. Xu and Y. Zailu, "Fuzzy model identification self-learning for dynamic system", *IEEE Trans on Systems, Man, Cybernetics*, vol. SMC-17, no.4, pp. 683-689, 1987.

[17] W. Pedrycz, "An identification algorithm in fuzzy relational system", *Fuzzy Sets and Systems*, vol. 13, pp. 153-167, 1984.

[18] Y. Lin and G. A. Cunningham Ⅲ, "A new approach to fuzzy-neural modeling", *IEEE Trans. Fuzzy Systems*, vol. 3, no. 2, 190-197, 1995.

Sung-Kwun Oh

He received the M.S. and Ph. D. degrees in electrical engineering from Yonsei University, Korea, in 1983 and 1993, respectively. He worked as a postdoctoral Fellow in the Department of Electrical and Computer Eng. Univ. of Manitoba, Canada, from 1996 to 1997. He is currently an Associate professor in the school of electrical, electronics and information engineering, Wonkwang University. His research interests include Fuzzy and Neural systems, intelligent control, and computational intelligence. He is currently a member of the KIEE, the ICASE, the KFIS, and the IEEE.

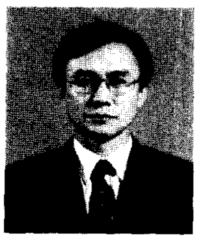


Dong-Won Kim

He was born in Chung-Joo, Korea, in 1974. He received the B.S. degree in control and instrumentation engineering from Wonkwang University, in 2000. He is currently studying for M.S. degree at the same institute. His research interests are mainly in the theory and technology of intelligent systems, control, and soft computing to learning and machine intelligence.

Byoung-Jun park

He received the B.S. and M.S. degrees in control and instrumentation engineering from the Wonkwang University, Korea in 1998 and 2000, respectively. He is currently a Ph. D. candidate at the same institute studying on a design of a fuzzy and neural networks. His research interests include Fuzzy and Neural systems, intelligent control, and computational intelligence.



Hyung-Soo Hwang

He received his M.S. and Ph. D. degree in electrical engineering from Chonbuk national University, in 1983 and 1987, respectively. He is currently a professor in the school of electrical, electronics and information engineering, Wonkwang University. His current research interests include Fuzzy control, Intelligent control, and Discrete Event Systems Control. He is currently a member of the ICASE, the IEEE, and the IEEK.