Calculation of the Power Peaking Factor Using CFNN and Its Uncertainty Analysis

Ju Hyun Back, Dong Yeong Kim, Kwae Hwan Yoo, Geon Pil Choi, and Man Gyun Na[*]
Department of Nuclear Engineering Chosun University, 309 Pilmun-daero, Dong-gu, Gwangju, Korea
[*]*Corresponding author: magyna@chosun.ac.kr*

1. Introduction

Fuel pellets and fuel clad barriers during reactor operation should be operated within the various safety limits such as the local power density (LPD) and the departure from nucleate boiling ratio (DNBR). Therefore, detailed 3-Dimensional (3D) core power distribution monitoring is required during reactor operation.

The LPD and DNBR must be calculated in order to perform the main functions of the core protection calculator (CPC) and the core operation limit supervisory system (COLSS) [1]. CPC and COLSS play a role in the protection and monitoring systems of the optimized power reactor 1000 (OPR1000) and the advanced power reactor 1400 (APR1400).

LPD should be estimated accurately to prevent fuel rods from melting. LPD at the hottest part of the core is called the power peaking factor (PPF, F_q). LPD at the hottest part of the core is more important than LPD at any other position in a reactor core. DNBR and PPF are the most important factors that must be continuously monitored from a safety aspect.

The aim of the study is to calculate PPF in a reactor core by a cascaded fuzzy neural networks (CFNN) model according to operating conditions. The CFNN can be used to optimize the value through the process of adding fuzzy neural networks (FNN) repeatedly. The operation condition is reactor power, core inlet temperature, pressurizer pressure, mass flowrate, axial shape index (ASI), and variety of control rod position.

The proposed CFNN model that is a PPF estimation algorithm is verified by using the nuclear and thermal data acquired from numerical simulations of OPR1000.

2. Methods and Results

2.1 Fuzzy inference system (FIS)

The fuzzy inference system (FIS) uses the conditional rules that is comprised of *if-then* rules of a pair of the antecedent and consequent [2]. This study uses the Takagi-Sugeno-type FIS [3], which does not need the defuzzifier in the output terminal because its output is a real value.

In the FIS, an arbitrary i^{th} fuzzy rule can be expressed as follows (first-order Takagi-Sugeno-type):

$$\begin{aligned}&\text{If } x_1(k) \text{ is } A_{i1}(k) \quad AND \cdots AND\, x_m(k) \text{ is } A_{im}(k),\\ &\text{then } y^i(k) \text{ is } f^i(x_1(k),\cdots,x_m(k))\end{aligned} \qquad (1)$$

where

$x_1,\cdots,x_m$: FIS input values

m = number of input variables

$A_{i1}(k),\cdots,A_{im}(k)$: fuzzy sets of the i^{th} fuzzy rule

y^i : output of the i^{th} fuzzy rule

$$f^i(x_1(k),\cdots,x_m(k)) = \sum_{j=1}^{m} q_{ij}x_j(k) + r_i \qquad (2)$$

q_{ij} : weight of the i^{th} fuzzy input variable

r_i : bias of the i^{th} fuzzy rule

Because the function $f^i(x(k))$ is expressed as the first-order polynomial of input variables, FIS is called the first-order Takagi-Sugeno-type FIS in Eq. (2). The number of N input and output training data $z^T(k) = \left(\mathbf{x}^T(k), y(k)\right)$ (where $\mathbf{x}^T(k) = (x_1(k), x_2(k), \cdots, x_m(k))$ and $k = 1, 2, \cdots, N$) are assumed to be available, and the input and output variables are normalized. In general, there is no special restriction on the shape of the membership functions. In this study, the symmetric Gaussian membership function is used to reduce the number of parameters to be optimized.

$$A_{ij}(x_j(k)) = e^{-(x_j(k)-c_{ij})^2 / 2s_{ij}^2} \qquad (3)$$

The FIS output $\hat{y}(k)$ is calculated by weight-averaging the fuzzy rule output $\hat{y}(k)$ as follow:

$$\hat{y}(k) = \sum_{i=1}^{n} \bar{w}^i(k)y^i(k) = \sum_{i=1}^{n} \bar{w}^i(k) f^i(\mathbf{x}(k)) = \mathbf{w}^T(k)\mathbf{q} \qquad (4)$$

where

$$\bar{w}^i(k) = \frac{w^i(x(k))}{\sum_{i=1}^{n} w^i(x(k))} \qquad (5)$$

$$w^i(k) = \prod_{j=1}^{m} A_{ij}(x_j(k)) \qquad (6)$$

n : number of fuzzy rules

$\mathbf{q} = [q_{11}\cdots q_{n1}\cdots\cdots q_{1m}\cdots q_{nm} r_1 \cdots r_n]^T$

$$\mathbf{w}(k) = [\bar{w}^i(k)x_1(k)\cdots\bar{w}^n(k)x_1(k)\cdots\cdots$$
$$\bar{w}^1(k)x_m(k)\cdots\bar{w}^n(k)x_m(k)\;\bar{w}^1(k)\cdots\bar{w}^n(k)]^T$$

The vector $\mathbf{q}$ is called a consequent parameter vector that has $(m+1)n$ dimensions, and the vector $\mathbf{w}(k)$ consists of input data and membership function values. The estimated output for a total of input and output data pairs induced from Eq. (4) can be expressed as follows:

$$\hat{\mathbf{y}} = \mathbf{W}\mathbf{q} \qquad (7)$$

where

$$\hat{\mathbf{y}} = [\hat{y}(1)\;\hat{y}(2)\cdots\hat{y}(N)]^T$$
$$\mathbf{W} = [\mathbf{w}(1)\;\mathbf{w}(2)\cdots\mathbf{w}(N)]^T$$

The matrix $\mathbf{W}$ has $N\times(m+1)n$ dimensions.

2.2 FIS training

In this study, the FIS is optimized using the two combined methods of a genetic algorithm and a least squares method. The training data were used to develop the FNN model. The test data were used to verify the developed FNN model, and they are different from the training data set. The following fitness function for the genetic algorithm is proposed to minimize the maximum error and root mean square (RMS) error.

$$F = \exp(-\lambda_1 E_1 - \lambda_2 E_2) \qquad (8)$$

where

$$E_1 = \sqrt{\frac{1}{N_t}\sum_{k=1}^{N_t}(y(k)-\hat{y}(k))^2}$$
$$E_2 = \max_k(y(k)-\hat{y}(k))$$

λ_1 : weighting value of RMS error

λ_2 : weighting value of maximum error

N_t : number of training data

The variable $y(k)$ is the actual output value and $\hat{y}(k)$ is its value estimated using the FNN model. If the antecedent parameters are determined using a genetic algorithm through selection, crossover, and mutation, the resulting parameters appear similar to Eq. (7) as a first-order combination. Therefore, the consequent parameter $\mathbf{q}$ can be calculated easily using the least squares method. That is, the consequent parameter $\mathbf{q}$ is calculated to minimize an objective function. The objective function consists of the square error between the actual value $y(k)$ and its estimated value $\hat{y}(k)$, and it is expressed as follows:

$$J = \sum_{k=1}^{N_t}(y(k)-\hat{y}(k))^2 = \sum_{k=1}^{N_t}(y(k)-w^T(k)q)^2$$
$$= \frac{1}{2}(\mathbf{y}_t - \hat{\mathbf{y}}_t)^2 \qquad (9)$$

where

$$\mathbf{y}_t = [y(1)\;y(2)\cdots y(N_t)]^T$$
$$\hat{\mathbf{y}}_t = [\hat{y}(1)\;\hat{y}(2)\cdots\hat{y}(N_t)]^T$$

A solution for minimizing the above objective function can be obtained using the following equation:

$$\mathbf{y}_t = \mathbf{W}_t\mathbf{q} \qquad (10)$$

where

$$\mathbf{W}_t = [\mathbf{w}(1)\;\mathbf{w}(2)\cdots\mathbf{w}(N_t)]^T$$

The matrix $\mathbf{W}_t$ has $N_t\times(m+1)n$ dimensions in Eq. (10). The parameter vector $\mathbf{q}$ can be solved easily from the pseudo-inverse as follows:

$$\mathbf{q} = (\mathbf{W}_t^T\mathbf{W}_t)^{-1}\mathbf{W}_t^T\mathbf{y}_t \qquad (11)$$

The parameter vector $\mathbf{q}$ can be calculated from a series of input and output data pairs and their membership function values because the matrix $\mathbf{W}_t$ consists of input data and membership function values.

2.3 Cascaded fuzzy neural networks (CFNN)

The foregoing FNN is composed of the fuzzy logic and neural network theory. Most of the existing FNN models have been proposed to implement different types of single-stage fuzzy reasoning mechanisms. However, single-stage fuzzy reasoning is only the most simple among a human being's various types of reasoning mechanisms. Syllogistic fuzzy reasoning, where the consequence of a rule in one reasoning stage is passed to the next stage as a fact, is essential to effectively build up a large scale system with high level intelligence [4]. Therefore, it is described by applying these techniques in this paper.

The CFNN model contains two or more inference stages where each stage corresponds to a single-stage FNN module. The architecture of the CFNN is shown in Fig. 1.

The CFNN can be used to estimate the target value through the process of adding FNN repeatedly. The second stage FNN uses the initial input variables and the output variable of the first stage FNN as input variable. Therefore, this process is repeated L times to find the optimum value if over-fitting phenomena do not appear.

Similarly to Eq. (1), an arbitrary i^{th} rule of the CFNN can be expressed as Eq. (12):

$$\begin{array}{ll} \text{Stage 1} & \left[\begin{array}{l} If\ x_1(k)\ is\ A_{i1}^1(k)\ AND \cdots AND\ x_m(k)\ is\ A_{im}^1(k), \\ then\ \hat{y}_1^i(k)\ is\ f_1^i(x_1(k), \cdots, x_m(k)) \end{array}\right] \\ \text{Stage 2} & \left[\begin{array}{l} If\ x_1(k)\ is\ A_{i1}^2(k)\ AND \cdots AND\ x_m(k)\ is\ A_{im}^2(k) \\ \quad AND\ \hat{y}_1(k)\ is\ A_{i(m+1)}^2(k), \\ then\ \hat{y}_2^i(k)\ is\ f_2^i(x_1(k), \cdots, x_m(k), \hat{y}_1(k)) \end{array}\right] \\ & \qquad \vdots \\ \text{Stage } L & \left[\begin{array}{l} If\ x_1(k)\ is\ A_{i1}^L(k)\ AND \cdots AND\ x_m(k)\ is\ A_{im}^L(k), \\ \quad AND\ \hat{y}_1(k)\ is\ A_{i(m+1)}^L(k)\ AND \cdots AND\ \hat{y}_{(L-1)}(k)\ is\ A_{i(m+L-1)}^L(k), \\ then\ \hat{y}_L^i(k)\ is\ f_L^i(x_1(k), \cdots, x_m(k), \hat{y}_1(k), \cdots, \hat{y}_{(L-1)}(k)) \end{array}\right] \end{array} \tag{12}$$

where L is the stage number of CFNN. The CFNN model is trained sequentially at each FNN module in the same way as explained in subsection 2.2 and 2.3

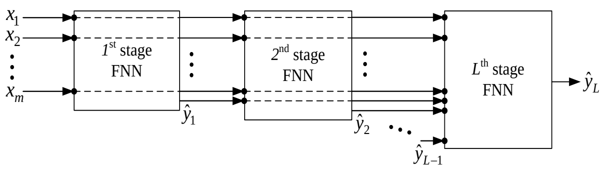


Fig. 1. Cascaded fuzzy neural network (CFNN)

2.4 Uncertainty Analysis

When CFNN models are used to estimate safety-critical parameters such as the LPD, model estimates require an uncertainty analysis to determine the accuracy of the data-based model prediction. Through an uncertainty analysis, a prediction interval can be calculated such that the exact value exists in the prediction interval at a specified confidence level.

There are several possible sources of uncertainty in predictions using data-based models; selection of training data, model structure including complexity, and noise in the input variables and the output variables [5]. Data-based model is developed using a given training data set. Each possible training data set selected from the entire population of data will generate a different model and there will be a distribution of predictions for a given observation data. Also, inappropriate model will cause a bias.

In this study, we analyzed the uncertainty about the CFNN model that has been developed using a statistical method. The bootstrap method works by generating many bootstrap samples of the training data set and re-training the CFNN model parameters on each bootstrap sample. After repetitive sampling and re-training, the resulting predictions provide a distribution for the LPD value. This distribution can be used to calculate prediction intervals. In this study, we used one of the pair of bootstrap sampling algorithm of statistical uncertainty analysis. The available data is divided into development data and test data. The development data consists of a large pool of data from which training and verification samples can be drawn. The test data is fixed. Uncertainty is separated into two types such as the variability and bias. The calculation steps of the bootstrap pairs sampling algorithm are as follows [6]:

At first, J samples ($J = 100$ in this paper) are generated from the development data, each one of size N drawn with replacement from the N training data $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)\}$. The j^{th} sample is denoted by $\{(\mathbf{x}_1^j, y_1^j), (\mathbf{x}_2^j, y_2^j), \cdots, (\mathbf{x}_N^j, y_N^j)\}$.

For each bootstrap sample, an FNN model is obtained. The variance and the bias of the i^{th} predicted value are estimated by

$$Var(\hat{y}_i) = \frac{1}{J-1} \sum_{j=1}^{J} \left[\hat{y}_i^j - \bar{\hat{y}}_i \right]^2$$

where $\bar{\hat{y}}_i = \frac{1}{J} \sum_{j=1}^{J} \hat{y}_i^j$

$$bias = \left\{ \frac{1}{n} \sum_{i=1}^{n} \frac{1}{J} \sum_{j=1}^{J} \left[\hat{y}_i^j - y_i^j \right]^2 \right\}^{1/2}$$

where n is the number of the development data.

The pool of development data represents all available data, excluding the specified set of fixed test data. Since bias estimates based on the training data can be much lower than bias estimates based on an independent set of data, especially in case of an overfitting, one should compute bias estimates based on the data pool rather than the training data. The estimate with a 95% confidence interval for an arbitrary test input $\mathbf{x}_o$ is

$$\hat{y}_0 \pm 2\sqrt{Var(\hat{y}_0) + bias^2} = \hat{y}_0 \pm \delta$$

3. Application to PPF estimation.

The proposed algorithm was applied to the first fuel cycle of OPR1000. The data was obtained by running the MASTER [7] and COBRA codes [8]. The MASTER (Multipurpose Analyzer for Static and Transient Effects of Reactor) reactor analysis code developed by KAERI (Korea Atomic Energy Research Institute) is a nuclear analysis and design code which can simulate the PWR and BWR core in 1-, 2-, 3-dimensional geometry. The MASTER code was designed to have a variety of capabilities such as static core design, transient core analysis and operation support and is interfaced with the COBRA code for thermos-hydraulic calculations.

The data obtained from simulation of the MASTER code comprise a total of 18816 input-output data pairs ($x_1, x_2, \cdots, x_9, y_r$) that can describe the reactor core states appropriately in the ranges of the input variables.

In this study, the used PPF data were composed of 100 pieces of test data. x_1 through x_9 are the input signals that represent the reactor power, core inlet temperature, coolant pressure, mass flowrate, axial shape index (ASI), R2, R3, R4 and R5 control rod positions, and y_r is the output signal which indicates the PPF in the reactor core. ASI is defined as $(P_B - P_T)/(P_B + P_T)$ where P_B is the bottom-half power and P_T is the top-half power of a nuclear reactor.

The two respective CFNN models were optimized for two kinds of data sets, the positive (relatively high power at the top part of the reactor core) ASI cases (7625 data points) and the negative ASI cases (7625 data points). This results in smaller errors compared with using only one summed data set. All the acquired data of the positive or negative ASI cases are divided into a training data set, a verification data set, and a test data set.

Table I shows the calculation results when the SPND signals were not used. The root mean squared (RMS) error is 0.0337% and the maximum error is 0.3368% for all data except the test data with positive ASI. The RMS error is 0.0485 and the maximum error 0.6471% for all data except the test data with negative ASI.

Table I: PPF Calculation Results by the CFNN

Without SPND signals		No. of data points	Relative maximum error (%)	RMS error (%)
Positive ASI	Development data	7525	0.1994	0.0314
	Test data	100	0.2963	0.0450
	All data except test data	7625	0.3368	0.0337
Negative ASI	Development data	7525	0.2856	0.0427
	Test data	100	0.3190	0.0597
	All data except test data	7625	0.6471	0.0485

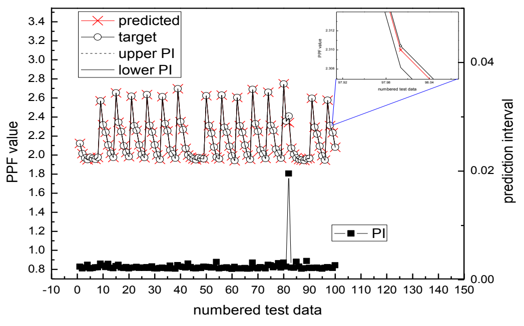
Fig. 2 shows the results of the uncertainty analysis for the CFNN PPF predictive model.

Actual PPF value and predictive PPF value exist in both upper prediction interval (PI) and Lower PI bound. Therefore, it can be seen that the prediction is reliable. PI is approximately below 0.01. Therefore, the PPF can be predicted very accurately.

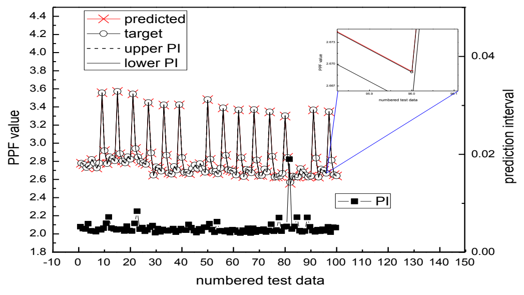
4. Conclusions

In this study, CFNN models have been developed and applied to the calculation of the PPF in the reactor core. The CFNN regression models were optimized by using the data set prepared as training data and tested by using verification data.

The developed CFNN models were applied to the OPR 1000. As a result, the RMS error of the estimated PPF values is below 0.05%. In addition, their uncertainty was analyzed by a bootstrap method using 100 sampled development data sets. The prediction intervals are very small, which means that the predicted values are very accurate. As a result of this study, the CFNN models are sufficiently accurate to be used in PPF monitoring.



(a) Negative ASI



(b) Positive ASI

Fig. 2. Uncertainty analysis for PPF predictive model.

REFERENCES

[1] ABB Combustion Engineering Inc, Overview Description of the Core Operation Limit Supervisory System (COLSS), CEN-312-P, Revision 01-P, 1986.
[2] E. H. Mamdani and S. Assilian, An experiment in linguistic synthesis with a fuzzy logic controller, Int. J. Man-Machine Studies, vol. 7, pp. 1-13, 1975.
[3] T. Takagi and M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, IEEE Trans. Systems, Man, Cybern., vol. SMC-1, no. 1, pp. 116-132, Jan./Feb. 1985.
[4] J.C. Duan and F.L. Chung, Cascaded fuzzy neural network model based on syllogistic fuzzy reasoning, IEEE Trans. Fuzzy Systems, vol. 9, no. 2, pp. 293-306, Apr. 2001.
[5] J.W. Hines, B. Rasmussen, Online sensor calibration monitoring uncertainty estimation, Nuclear Technology, vol. 151, pp. 281-288, Sept. 2005.
[6] R. Tibshirani, A comparison of some error estimates for neural network models, Neural Computation, vol. 8, pp. 152-163, 1996.
[7] B.O. Cho, H.G. Joo, J.Y. Cho, and S.Q. Zee., MASTER: reactor core design and analysis code, in Proc. 2002 Int. Conf. New Frontiers of Nuclear Technology: Reactor Physics (PHYSOR 2002), Seoul, Korea, Oct. 7-10, 2002.
[8] C.L. Wheeler, C.W. Stewart, R.J. Cena, D.S. Rowe, and A.M. Sutey, COBRA-IV-I; An interim version of COBRA for thermal hydraulic analysis of rod bundle nuclear fuel elements and cores, BNWL-1962, March 1976.