Melt Index Prediction Based on Fuzzy Neural Networks and PSO Algorithm with Online Correction Strategy

Xinggao Liu and Chengye Zhao
State Key Laboratory of Industrial Control Technology, Dept. of Control Science and Engineering, Zhejiang University, Hangzhou 310027, P.R. China

DOI 10.1002/aic.12660
Published online May 13, 2011 in Wiley Online Library (wileyonlinelibrary.com).

A black-box modeling scheme to predict melt index (MI) in the industrial propylene polymerization process is presented. MI is one of the most important quality variables determining product specification, and is influenced by a large number of process variables. Considering it is costly and time consuming to measure MI in laboratory, a much cheaper and faster statistical modeling method is presented here to predicting MI online, which involves technologies of fuzzy neural network, particle swarm optimization (PSO) algorithm, and online correction strategy (OCS). The learning efficiency and prediction precision of the proposed model are checked based on real plant history data, and the comparison between different learning algorithms is carried out in detail to reveal the advantage of the proposed best-neighbor PSO (BNPSO) algorithm with OCS. © 2011 American Institute of Chemical Engineers *AIChE J,* 58: 1194–1202, 2012
Keywords: fuzzy neural network, particle swarm optimization, melt index prediction, online correction strategy

Introduction

Production of polypropylene is a multi-billion business. Melt index (MI) is one of the most important parameters, which determines the grade of product and affects the control strategy of practical polypropylene produced. To get an intuitional understanding of the process, a highly simplified schematic diagram of the process is illustrated in Figure 1. This process consists of a chain of reactors in series, two continuous stirred-tank reactors and two fluidized-bed reactors. Propylene and hydrogen are fed into each reactor, but the catalyst is added only to the first reactor along with the solvent. These liquids and gases supply reactants for the growing polymer particles and provide the heat transfer media. The polymerization reaction takes place in a liquid phase in the first 2 reactors and is completed in vapor phase in the third and fourth reactors to produce the powdered polymer products. In general, the MI of the polypropylene depends on the catalyst properties, reactant composition, and reactor temperature, etc.

MI is usually evaluated off-line with an analytical procedure, which takes almost 2 h in the laboratory,[1] leaving the process without any real-time quality indicator and usually leads to off-specification products and profit losses. An alternative approach is to develop a dynamic model based on the mechanism of polymerization process[2–4] and certain easy-measurable process variables. However, this approach is often challenged by the engineering activity and the relatively high complexity of the kinetic behavior and operation of the polymer plants.[5,6] So developing of statistical model, which directly uses available process variables at the beginning of the production cycle to predicting MI of the final product, is more practical and convenient.

Data based methodologies for process monitoring, modeling, and control[7] have already been studies and applied in several engineering fields. Kinds of neural networks, thanks to their extremely powerful adaptive capabilities in response

Correspondence concerning this article should be addressed to X. Liu at lxg@zjuem.zju.edu.cn.

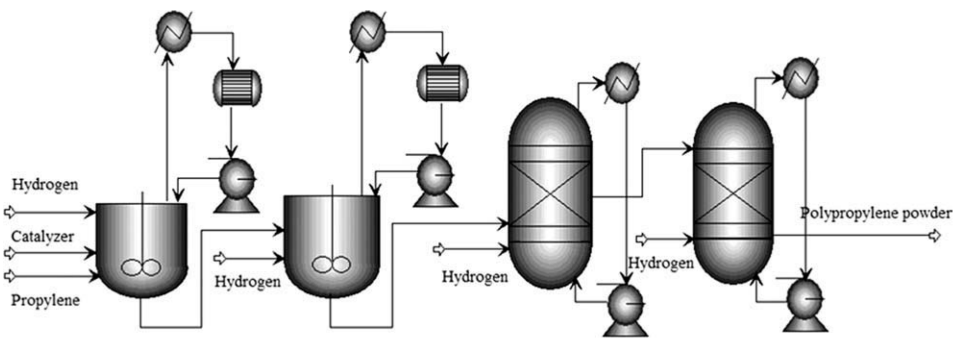

Figure 1. General scheme of propylene polymerization.

to nonlinear behaviors,[8,9] have been widely applied to model and control dynamic processes. Xiong and Zhang[10] presented a strategy to overcome the problems of unknown disturbances and model-plant mismatches in fed-batch process optimal control by using modified iterative dynamic programming algorithm. Han et al.[11] used three different approaches, including supported vector machines (SVM), partial least squares, and back propagation (BP) neural network, to estimate MI in propylene polymerization (PP) process. Shi and Liu[12,13] developed several soft-sensor models for MI prediction based on radial basis function (RBF) neural network, and weighted least squares SVM. Zhang et al.[14] proposed a sequential training method of neural networks, which was applied in the inferential estimation of the polymer MI in an industrial plant and obtained a quite good performance. Geng et al.[15] presented a dynamic soft-sensing model based on diagonal recurrent neural network to predict actual industrial process variable. Dua[16] proposed a mix-integer programming approach to automatically generate the optimal configuration of the network and to eliminate redundant nodes and interconnections, which was applied on practical fault diagnosis and prediction of compositions. Although these works have achieved better and better MI prediction accuracy, greater performance in prediction and better universality of the estimation model are still the first-line goal in academic and industrial community.

However, few literatures have mentioned the using of fuzzy method or fuzzy neural network (FNN) in developing statistical MI estimator. FNN is a hybrid system combining the theories of fuzzy logic and neural network, which can make effective use of easy interpretability of fuzzy logic as well as superior learning ability and adaptive capability of neural networks. It has wide applications in areas of adaptive control, adaptive signal processing,[17] nonlinear system identification,[18,19] pattern recognition, etc. The design of FNN model consists of structure and parameter identification. Parameter identification involves determining parameters of premises and consequences. Structure identification comprises the partitioning of input–output space and determination of the rule number for the desired performance. Various methods adopted for building FNN system in literatures, including adaptive-network-based fuzzy inference system (ANFIS), self-organizing FNN,[20] FNN using particle swarm optimization (PSO) and recursive SVD,[21] and FNN with group-based symbiotic evolution.[22]

PSO algorithm was first raised by Eberhart and Kennedy, which uses the behavior of natural animal such as bird flocking, fish schooling, and swarm theory to yield the best of the characteristics among the population.[23] PSO has become a popular optimizer and has been widely applied in solving practical problem, and its theoretical studies and performance improvements have become important and attractive. Much research has been reported, including parameter studies, topological structures, comprehensive PSO algorithm, and cooperative PSO algorithm.[24] A best-neighbor PSO (BNPSO) algorithm based on square topological structure is applied, which successfully overcomes the defects of the traditional PSO and achieves a perfect optimizing performance. The algorithm is further combined with online correcting strategy to optimize model configurations while running online.

In this article, "Model Architecture and Learning" section presents an overview of the structure and calculation of FNN as well as details of PSO algorithm and the online correcting strategy. "Melt Index Prediction" section presents the case study, including input variable selection, performance criterion selection, configurations of algorithm parameters, and the performance of the proposed model. Finally, conclusion and some directions for future research are discussed in "Conclusions" section.

Model Architecture and Learning

The architecture of the prediction model is shown in Figure 2. The architecture contains two major components, PSO-online correction strategy (OCS) learning module and FNN module. The PSO-OCS module involves batch learning and sequence learning, which used the history dataset to get the initial optimization configurations of FNN model off-line and use OCS to adaptively optimize configuration online, respectively. Details of these modules are presented as follow.

Fuzzy neural network

FNN consists of a set of fuzzy IF-THEN rules, which describe the input–output mapping relationship of the network. The antecedents of fuzzy rules partition the input space into a number of linguistic term sets while the consequent constituent can be chosen as a fuzzy membership function (Mamdani model), a singleton value,[25] or a function of a linear combination of input variables (Takagi-Sugeno Kang (TSK) model).[26] No matter which type of neural fuzzy networks is chosen, different consequent constituents result in different types of fuzzy models. Without loss of generality, multi-input single-output (MISO) fuzzy model is applied in this article with singleton consequent which consists of N rules:

$$R_j: \text{ IF } x_1 \text{ is } A_{1j} \text{ and } x_2 \text{ is } A_{2j} \dots \text{ and } x_n \text{ is } A_{nj} \quad \text{THEN } y = w_j,\ j = 1, 2, \dots, N \quad (1)$$

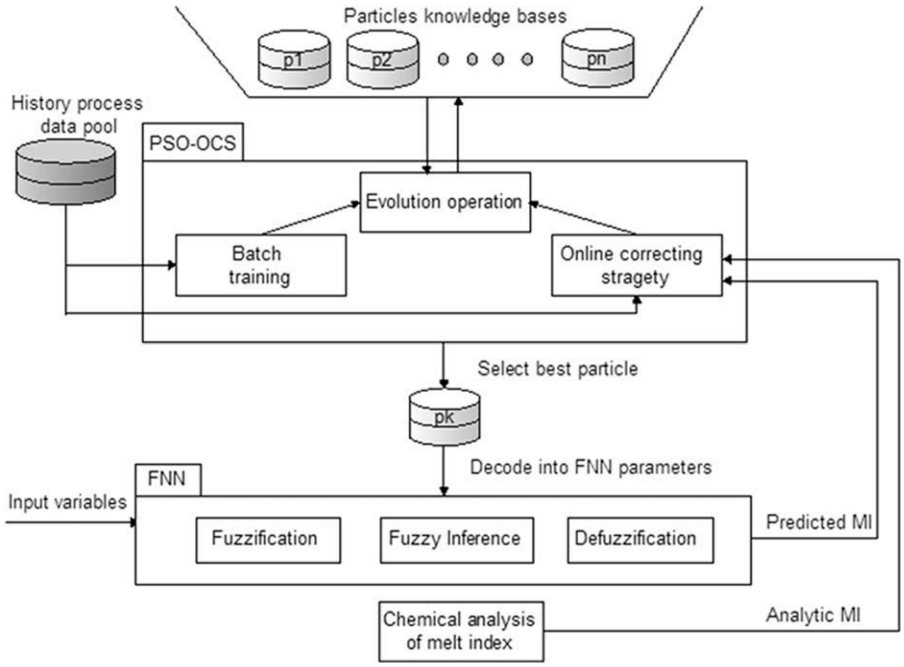

Figure 2. Melt index prediction system paradigm.

where x_i is the input variable, y is the output variable, A_{ij} is the linguistic term of the precondition part, w_j is the output action strength associated with the jth rule, N is the number fuzzy rules, and n is the number of input variables.

The structure of a four-layer FNN is shown in Figure 3. The functions of the nodes in each layer are described as follows:

Layer 1: Input layer. Each node in layer 1 represents an input variable. The node only transmits input values to layer 2.

$$u_i^{(1)} = x_i, \quad i = 1, 2, \ldots n \tag{2}$$

Layer 2: Fuzzification layer. Nodes in layer 2 are arranged into N groups, each group representing the if-part of a fuzzy rule. Each node computes the value of membership function. For an external input $u_i^{(1)}$, the following Gaussian membership function is used:

$$u_{ij}^{(2)} = \exp\left(-\frac{[u_i^{(1)} - m_{ij}]^2}{\sigma_{ij}^2}\right), \; i = 1, 2, \ldots n, \; j = 1, 2, \ldots, N \tag{3}$$

where m_{ij} and σ_{ij} are the center and the width of the Gaussian membership function, respectively, of the jth term of the ith input variable x_i.

Layer 3: Fuzzy inference layer. Product inference method is used in this article; the number of nodes in layer 3 is equal to the number of fuzzy rules. Each node computes the rule activation strength, for the jth rule R_j, its output is

$$u_j^{(3)} = \prod_i u_{ij}^{(2)}, \quad j = 1, 2, \ldots, N \tag{4}$$

Layer 4: Output layer. Nodes in this layer perform defuzzification. The mathematical function of each node is:

$$u^{(4)} = \frac{\sum_j u_j^{(3)} w_j}{\sum_j u_j^{(3)}} \tag{5}$$

where $u^{(4)}$ is the output of the FNN.

The first step of building FNN model is the transformation of fuzzy rules into a coded sequence. As shown in Figure 4, FNN with n inputs and N rules can be coded into a sequence of length $(2n + 1)N$. In each rule, m_{ij} and σ_{ij}, where i and j represent ith dimension and jth rule, represent a Gaussian membership function with mean m_{ij} and deviation σ_{ij}, respectively, and w_j represents the output weight of the jth rule node.

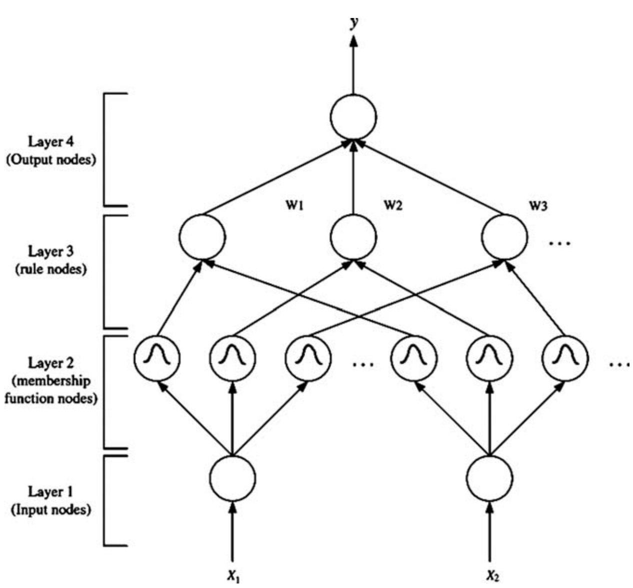

Figure 3. Structure of a fuzzy neural network.

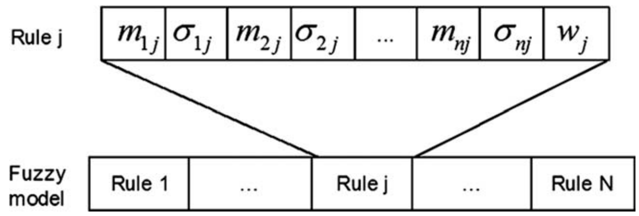

Figure 4. Coding a fuzzy neural network into a chromosome in PSO.

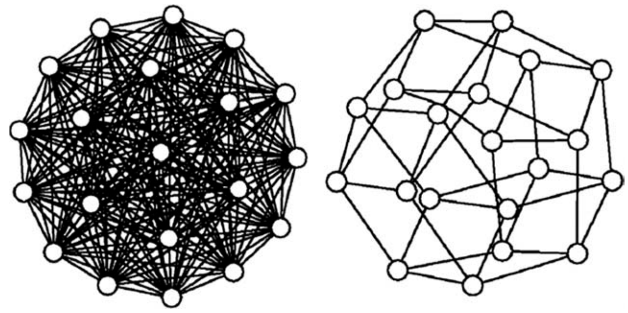

Figure 5. GPSO and BNPSO population sociometries.

We use PSO algorithm directly to optimize structure of FNN and the configuration of network can also be obtained automatically by using multiple rounds of running of PSO algorithm. We also use a more automatic clustering method, which can determine the number of clusters. By using self-clustering algorithm, the results show that the optimal FNN structure should contain five fuzzy rules and totally 95 parameters in this work.

PSO algorithm

In classical PSO algorithm, also named global PSO (GPSO), the Lbest is the best solution of the individual particle that it has achieved so far. The Gbest is obtained by choosing the overall best value from all particles. At each iteration step, the velocity and position of the particle is updated by the following equation:

$$v_{pk}(t+1) = \eta v_{pk}(t) + \varphi_1 \alpha_1 (\mathrm{Lbest}_{pk} - r_{pk}(t)) + \varphi_2 \alpha_2 (\mathrm{Gbest}_k - r_{pk}(t)), p = 1,2,...,P, k = 1,2,...,D \quad (6)$$

$$r_{pk}(t+1) = r_{pk}(t) + v_{pk}(t+1),\ p = 1,2,...,P, k = 1,2,...,D \quad (7)$$

where v_{pk} is the corresponding velocity of the pth particle in the dth dimension space and r_{pk} is the responding solution of the pth particle in the dth dimension. Here, P is the total number of particles, D is the dimensionality of the search space, t describes current state, $t + 1$ describes the next state, α_1 and α_2 are acceleration constant, φ_1 and φ_2 are random numbers uniformly distributed in [0,1], and η is the scaling factor to control the learning rate. As the velocity of the particle is determined, the particle's solution will be modified at the next time step $t + 1$.

In GPSO algorithm, the source of social influence on each particle is the best-performing individual in the entire population. This is equivalent to a social network where every individual is connected to every other one, as shown in Figure 5. In BNPSO algorithm,[27,28] individuals are typically included in their own neighborhoods. In BNPSO algorithm, the velocity and position for each particle are updated by the following equations:

$$v_{pk}(t+1) = \chi(v_{pk}(t) + \varphi_1 (Lbest_{pk} - r_{pk}(t)) + \varphi_2 (Nbest_k - r_{pk}(t)),\ p = 1,2,...,P, k = 1,2,...,D \quad (8)$$

$$r_{pk}(t+1) = r_{pk}(t) + v_{pk}(t+1),\ p = 1,2,...,P, k = 1,2,...,D. \quad (9)$$

where φ_1 and φ_2 are random numbers uniformly distributed in $[0, \frac{\varphi_{\max}}{2}]$, and χ is the constriction coefficient that causes convergence of the individual trajectory in the search space.

BNPSO offered the advantage that subpopulations could search diverse regions of the problem space. As some individuals in one part of the population influenced one another to focus on one local optimum, another part of the population could search around another. The flow of information from one part of the population to another was moderated by the necessity of "persuading" intermediate individuals to search in a particular area. Once their best performance was in that region, they could influence their neighbors, and influence could slowly travel through the neighborhood.

Online Correcting Strategy

The complexity and nondeterminacy of industrial process system has been well known, and this leads to the requirement of learning and adaptive abilities of dynamic model. So one of the biggest challenges we will face when apply the MI model for real use is to adjust the model to overcome the instability of the process system. However, configurations of static models such as RBF neural network and FNN cannot be adapted easily once they were trained. So although new training data can be obtained continually during the process, an effective online learning algorithm is necessary to retrain the predictive model.

In this article, a no-strict online correcting strategy (OCS) based on PSO algorithm is added to the MI prediction model. Batch training is still applied during the initial construction of FNN model, and OCS is activated only when new training sample is available during the process. Usually, only the global optimum point is reserved after the computation, whereas the information of all the other particles' position is discarded. In fact, when come across the complex and multimodal problems such as neural network training, all the particles are valuable to reflect the feature of solution space. The simple idea of OCS is to make full use of the reserved swarm information of the batching to accelerate the online training. Flow chart of OCS is shown in Figure 6. The analytic value of MI at time t can be available at time $t + d$ due to the delay of analysis. The analytic value MI_{a}, along with the predicted value MI_{p} value and input variables x at time t is then put into OCS. OCS first calculate the difference between MI_{a} and MI_{p}. For example, if $|MI_{\mathrm{a}} - MI_{\mathrm{p}}| >$ mxae, where mxae is the maximum absolute error the FNN model performances on train dataset, a new train data $\{x(t), MI_{\mathrm{a}}(t)\}$ should be added into the train dataset. BNPSO algorithm starts using the new train dataset to get a new FNN

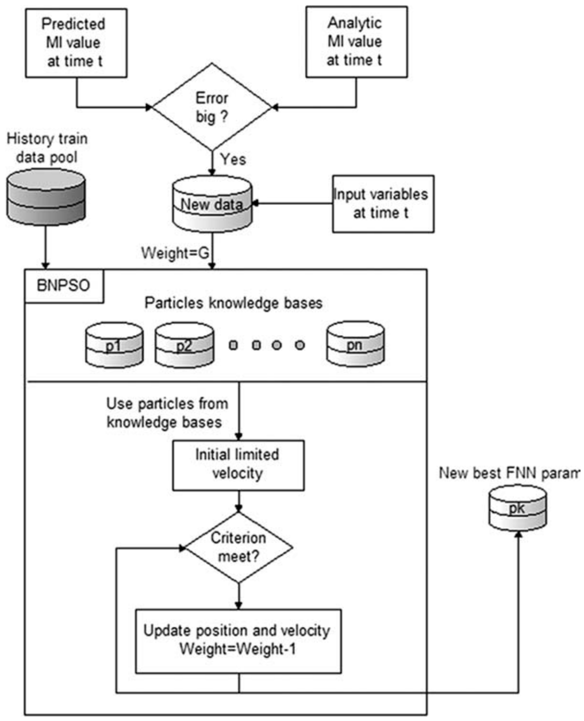

Figure 6. Flow chart of online correcting strategy.

model. In this progress, the velocity of particle is limited to enhance stability, and the new added dataset takes a large weight at the beginning, which decreases linearly through iterations.

Melt Index Prediction

The history data used for training and testing the prediction model are retrieved from the historical logs recorded in a real propylene polymerization plant. Data are filtered to discard abnormal situations and to improve the quality of the prediction system. Totally, nine process variables (t, p, l, a, $f1$, $f2$, $f3$, $f4$, $f5$) have been chosen to develop the MI prediction model, where t is the process temperature, p is the pressure, l is the level of liquid, a is the percentage of hydrogen in vapor phase, $f1$–$f3$ are flow rates of three streams of propylene, and $f4$ is the flow rate of catalyst as well as $f5$ is the flow rate of aid catalyst, respectively. A collection of 170 pairs of input–output data are used in this research. It is noted that the test set is obtained from the same batch as the training set, whereas the generalization set is derived from another batch.

This data collection are divided into three set, the train, test, and generalization sets, and 60 of which are used as the train dataset, 60 are used as the test set, and the rest of the data points form generalization set. All these original input and output data are normalized linearly into range [0,1] such that the direct output of prediction model should be denormalized to get the true MI value.

The following measurements of prediction error are used in this article, including the mean absolute error (MAE), the mean relative error (MRE), the root of mean square error (RMSE), and the Theil's inequality coefficient (TIC). The error indicators are defined as following:

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{10}$$

$$\mathrm{MRE} = \frac{1}{N}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{y_i} \times 100\% \tag{11}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{12}$$

$$\mathrm{TIC} = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}}{\sqrt{\sum_{i=1}^{N}y_i^2} + \sqrt{\sum_{i=1}^{N}\hat{y}_i^2}} \tag{13}$$

where y_i and $\hat{y}_i$ denote the measured value and predicted result, respectively. Different FNN models are compared by their performance on test and generalization dataset, because almost all of them perform well on train dataset. The average number of iteration each algorithm takes to meet the stop criterion in batch train stage is also measured as following:

$$\mathrm{ANI} = \frac{\sum_{i}^{RN} NI_i}{RN} \tag{14}$$

In this research, every algorithm is repeated $RN = 10$ times to measure its accuracy and robustness, and NI_i is iteration times in ith experiment.

Three different algorithms are implemented to test their performance on FNN prediction model design:

1. GPSO is applied. In this situation, a Sugeno-type zeros-order FNN with five rules is randomly initialized. The particle number P is set to 100, and constants η, α_1, and α_2. are initialized to 0.75, 1.2, and 1.2, respectively. The initial value of velocity vectors is constrained into [−0.5,0.5]. The algorithm stops when the mean squared error performance meets 10^{-4} or the iteration times exceed 1000.
2. In BNPSO algorithm, a Sugeno-type zeros-order FNN with five rules is randomly initialized first. The particle number P is set to 100, and constant χ takes value of 0.7298 as well as $\varphi_{\max}$ is initialized as 4. The initial value of velocity vectors is constrained into [−0.5,0.5]. The algorithm stops

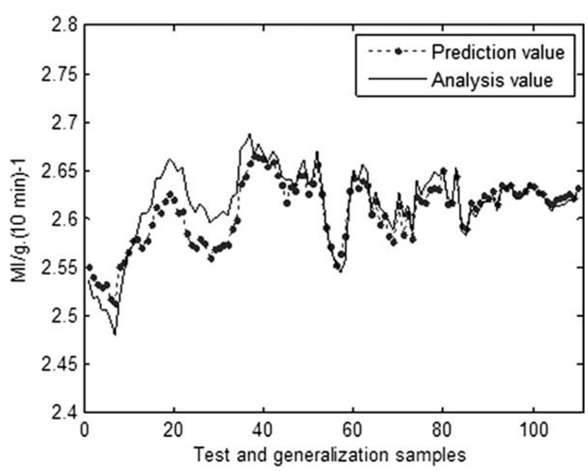

Figure 7. Performance of the GPSO-FNN model on test and generalization dataset.

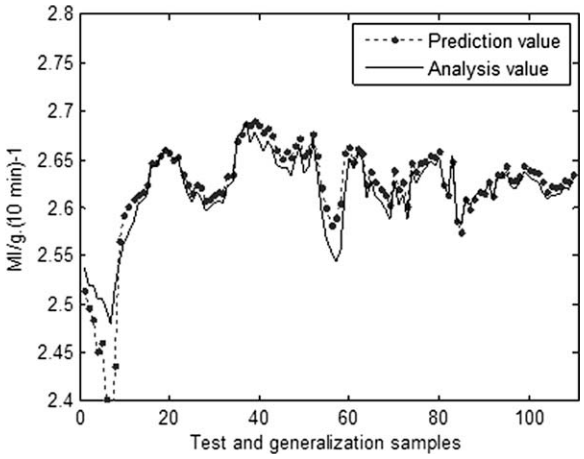

Figure 8. Performance of the BNPSO-FNN model on test and generalization dataset.

when the mean squared error performance meets 10^{-4} or the iteration times exceed 1000.

3. In BNPSO with OCS (BNPSO-OCS), the same parameter settings as BNPSO are used in batch training stage. In OCS stage, the initial value of velocity vectors is constrained into $[-0.05, 0.05]$ for stability, and the max iteration number is set to 50.

The performance of the best GPSO-FNN model on test and generalization dataset is shown in Figure 7. It is shown that the GPSO-FNN model is bad on test dataset but has a good performance on generalization dataset. The experiment also shows that GPSO takes a relative low probability (about 1 of 10) to get a FNN model works well both on train dataset and test dataset. This is because GPSO will easily converge into a local optimal solution.

Figure 8 illustrates the performance of the best BNPSO-FNN model on test and generalization dataset. The BNPSO-FNN model works well on both test and generalization dataset except rare data points, such as data points 53–58. The disadvantage of BNPSO is that it converges slower than GPSO because the evolution information is transformed through local neighbor, and in GPSO, the information is broadcast in whole swarm. Figure 9 illustrates the performance of the BNPSO-OCS-FNN model on test and generalization dataset. The

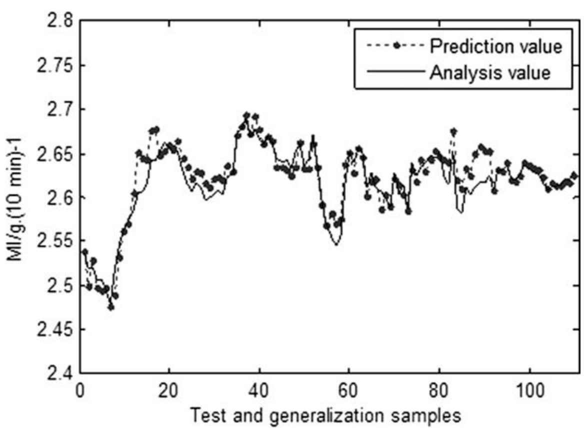

Figure 9. Performance of the BNPSO-OCS-FNN model on test and generalization dataset.

Table 1. Performances Comparison Between BNPSO and BNPSO-OCS on Test Dataset

Performance	ANFIS	GPSO-FNN	BNPSO-FNN	BNPSO-OCS-FNN
MAE	0.025	0.0140	0.0040	**0.0031**
MRE (%)	0.998	0.5346	0.1565	**0.1207**
RMSE	0.036	0.0155	0.0060	**0.0053**
TIC	0.0068	0.0029	0.0012	**0.0010**
AIN	–	**141.50**	495.90	495.90

BNPSO-OCS-FNN model precisely predicts most of the test and generalization dataset due to its online adjust mechanism.

Table 1 shows that the BNPSO-OCS-FNN model functions best overall on the testing dataset. In details, GPSO-FNN model gives an MAE of 0.0140, an MRE of 0.5346%, a RMSE of 0.0155, and a TIC of 0.0029. BNPSO-FNN model obtains an MAE of 0.0040, an MRE of 0.1565%, a RMSE of 0.0060, and a TIC of 0.0012. BNPSO-FNN model has already obtained improved prediction accuracy than the GPSO-FNN model. However, the BNPSO-OCS-FNN model achieves even better performance. The MAE, MRE, RMSE, and TIC of this model are 0.0031, 0.1207%, 0.0053, and 0.0010, with percentage decreases of 77.86%, 77.42%, 61.29%, and 58.62% compared with those of the GPSO-FNN model, respectively. These error indicators prove the BNPSO-OCS-FNN model provides wonderful MI prediction accuracy for the propylene polymerization process.

To specify the universality of the proposed method, models are also evaluated on the generalization dataset. An accurate prediction of MI on this dataset gives a strong support that the model owns good universality. Table 2 lists the specific error indexes for GPSO-FNN model, BNPSO-FNN model, and BNPSO-OCS-FNN model when they predict on generalization dataset. In terms of MAE, RMSE, and TIC, the BNPSO-FNN model still win over GPSO-FNN model, but lose to BNPSO-OCS-FNN model. Clearly, BNPSO-OCS-FNN model gives a nearly real MI value prediction, more accuracy than GPSO-FNN model and BNPSO-FNN model do. In Tables 1 and 2, the prediction results of FNN model based on PSO algorithm are further compared with ANFIS model, and it is obviously that the prediction performances of ANFIS model are not so satisfied on test dataset and generalization dataset although the same model performance perfect on train dataset. Table 3 compares the BNPSO-OCS-FNN model with other models presented in the open literatures. Our work improve the prediction precise from MRE 0.84%, presented in Ref. 13 and being the best result in literatures so far, to MRE 0.12%, which indicates that the method present in this article has obviously improved the prediction precise.

Table 2. Performances Comparison Between GPSO, BNPSO, and BNPSO-OCS on Generalization Dataset

Performance	ANFIS	GPSO-FNN	BNPSO-FNN	BNPSO-OCS-FNN
MAE	0.0337	0.0153	0.0016	**0.0015**
MRE (%)	1.2877	0.5854	0.0601	**0.0583**
RMSE	0.0345	0.0155	**0.0017**	**0.0017**
TIC	0.0065	0.0029	**0.0003**	**0.0003**
AIN	–	**141.50**	495.90	495.90

Table 3. **Performances Comparison Between BNPSO-OCS and Other Models Presented in the Open Literatures**

Literatures	Model	MRE	RMSE
This paper	BNPSO-OCS-FNN	0.12%	0.0053
Cao[29]	Adaptive RBF	–	0.62
Shi and Liu[13]	ICA-MS-RBF	2.98%	0.0794
Lou and Liu[30]	PCA-GA-RBF	0.84%	–

In general, this research mainly focuses on presenting a MI model with online fast learning feature, which solves the online training efficiency, a bottleneck problem when the model is applied practically. However, future research on MI modeling may involve systematic method[16] to further improve the efficiency and precision of our model.

Conclusions

This article presents a statistical modeling method to develop an online MI prediction model, which aims to replace costly and time-consuming analytical measurement in laboratory. The work combines technologies of the FNN, PSO algorithm, and online correcting strategy (OCS). The MISO FNN is applied to form the basic architecture of the prediction model, which expresses the unknown relationship between input process variables and output MI. Three different learning algorithms, including GPSO algorithm, BNPSO algorithm, and BNPSO with online correcting strategy (OCS) have been applied to optimize parameters of FNN. BNPSO algorithm, which uses square topological structure instead of global structure in GPSO, shows advantage of high convergence speed and optimization precision than GPSO algorithm. Furthermore, the application of OCS improves the performance of the prediction model on test dataset and generalized dataset. Research work proves the reliability and efficiency of the BNPSO algorithm and indicates that the proposed FNN model is capable of express the relationships between process variables measured at the beginning of the production cycle and the quality of the final product as well as shows the proposed modeling approach is supposed to have a promising potential for practical use.

Acknowledgments

This work is supported by National Natural Science Foundation of China (Grant 50876093), International Cooperation and Exchange Project of Science and Technology Department of Zhejiang Province (Grant 2009C34008), and Zhejiang Provincial Natural Science Foundation for Distinguished Young Scientists (Grant R4100133) as well as their supports are thereby acknowledged.

Literature Cited

1. Bafna SS, Beall AM. A design of experiments study on the factors affecting variability in the melt index measurement. *J Appl Polym Sci*. 1997;65:277–288.
2. Reginato AS, Zacca JJ, Secchi AR. Modeling and simulation of propylene polymerization in nonideal loop reactors. *AIChE J*. 2003;49: 2642–2288.
3. McKenna TF, Soares JBP. Single particle modeling for olefin polymerization on supported catalysts: a review and proposals for future developments. *Chem Eng Sci*. 2001;56:3931–3949.
4. Luo ZH, Shi DP, Zhu Y. Multiple active site Monte Carlo model for heterogeneous Ziegler-Natta propylene polymerization. *J Appl Polym Sci*. 2009;115:2962–2970.
5. Lee EH, Kim TY, Yeo YK. Prediction and quality control of the melt index during production of high-density polyethylene. *Korean J Chem Eng*. 2008;25:613–622.
6. Luo ZH, Zheng Y, Cao ZK, Wen SH. Mathematical modeling of the molecular weight distribution of polypropylene produced in a loop reactor. *Polym Eng Sci*. 2007;47:1643–1649.
7. Ahmed F, Nazir S, Yeo YK. A recursive PLS-based soft sensor for prediction of the melt index during grade change operation in HDPE plant. *Korean J Chem Eng*. 2009;26:14–20.
8. Chaudhuri BB, Bhattacharya U. Efficient training and improved performance of multilayer perceptron in pattern classification. *Neurocomputing*. 2000;34:11–27.
9. Hunt KJ, Sbarbaro D, Zbikowski R, Gawthrop PJ. Neural networks for control systems—a survey. *Automatica*. 1992;28:1083–1112.
10. Xiong ZH, Zhang J. Neural network model-based on-line re-optimisation control of fed-batch processes using a modified iterative dynamic programming algorithm. *Chem Eng Process*. 2005;44:477–484.
11. Han IS, Han C, Chung CB. Melt index modeling with support vector machines, partial least squares and artificial neural networks. *J Appl Polym Sci*. 2005;95:967–974.
12. Shi J, Liu XG. Melt index prediction by weighted least squares support vector machines. *J Appl Polym Sci*. 2006;101:285–289.
13. Shi J, Liu XG. Melt index prediction by neural networks based on independent component analysis and multi-scale analysis. *Neurocomputing*. 2006;70:280–287.
14. Zhang J, Jin Q, Xu Y. Inferential estimation of polymer melt index using sequentially trained bootstrap aggregated neural networks. *Chem Eng Technol*. 2006;29:442–448.
15. Geng H, Xiong ZH, Mao S, Xu YM. Dynamic soft-sensing model by combining diagonal recurrent neural network with levinson predictor. *Lect Notes Comput Sci*. 2006;3973:1059–1064.
16. Dua V. A mixed-integer programming approach for optimal configuration of artificial neural networks. *Chem Eng Res Des*. 2010;88:55–60.
17. Juang CF, Lin CT. An on-line self-constructing neural fuzzy inference network and its applications. *IEEE T Fuzzy Syst*. 1998;6:12–32.
18. Mastorocostas PA, Theocharis JB. A recurrent fuzzy neural model for dynamic system identification. *IEEE Trans Syst Man Cybern B Cybern*. 2002;32:176–190.
19. Wu S, Er MJ. Dynamic fuzzy neural networks-a novel approach to function approximation. *IEEE Trans Syst Man Cybern B Cybern*. 2000;30:358–364.
20. Qiao JF, Wang HD. A self-organizing fuzzy neural network and its applications to function approximation and forecast modeling. *Neurocomputing*. 2008;71:564–569.
21. Lin CJ, Hong SJ. The design of neuro-fuzzy networks using particle swarm optimization and recursive singular value decomposition. *Neurocomputing*. 2007;71:297–310.
22. Lin CJ, Xu YJ. A self-adaptive neural fuzzy network with group-based symbiotic evolution and its prediction applications. *Fuzzy Set Syst*. 2006;157:1036–1056.
23. Eberhart RC, Shi YH. *Particle swarm optimization: developments, applications and resources*. In: *Proceedings of the Congress on Evolutionary Computation*, Seoul, Korea, 2001: 81–86.
24. Bergh FVD, Engelbrecht AP. A cooperative approach to particle swarm optimization. *IEEE Trans Evol Comput*. 2004;8:225–239.
25. Lin FJ, Lin CH, Shen PH. Self-constructing fuzzy neural network speed controller for permanent-magnet synchronous motor drive. *IEEE Trans Fuzzy Syst*. 2001;9:751–759.
26. Castellano G, Fanelli AM, Mencar C. A neuro-fuzzy network to generate human-understandable knowledge from data. *Cogn Syst Res*. 2002;3:125–144.
27. Kennedy J, Mendes R. Neighborhood topologies in fully informed and best-of-neighborhood particle swarm. *IEEE Trans Syst Man Cybern C Appl Rev*. 2006;36:515–519.
28. Kennedy J, Mendes R. *Population structure and particle swarm performance*. In: *Proceedings of the World Congress on Computational Intelligence*, Los Angeles, USA, 2002: 1671–1676.
29. Cao J, Wang GZ, Xu BW. Prediction of polypropylene melt index based on robust and adaptive RBF networks. *Control Decis*. 1999; 14:341–343.
30. Lou W, Liu XG. Melt index prediction of polypropylene by neural networks model based on PCA-GA-RBF. *J Petrochem Univ*. 2007; 20:82–85.

Appendix: Model Test and Generalization Dataset

Test dataset

t	p	l	a	$f1$	$f2$	$f3$	$f4$	$f5$	MI
70.01	29.511	39.47	0.166	10.02	3800	3996	64.10	28.029	2.5735
70.00	29.498	39.50	0.174	9.48	3799	3636	63.88	27.977	2.5864
69.95	29.442	39.61	0.184	9.52	3800	2935	63.43	27.939	2.6052
70.03	29.468	39.56	0.191	9.76	3800	2933	60.06	27.982	2.6070
70.16	29.567	39.64	0.203	9.61	3799	2920	56.41	27.975	2.6142
70.07	29.500	39.64	0.226	9.54	3800	2879	59.50	28.065	2.6432
70.01	29.485	39.65	0.223	9.53	3800	2862	59.52	28.120	2.6415
70.05	29.528	39.59	0.234	9.24	3799	2850	59.49	28.009	2.6520
70.07	29.550	39.55	0.244	9.80	3800	2876	59.72	27.910	2.6619
70.06	29.551	39.64	0.239	10.03	3800	2841	59.80	27.873	2.6579
70.02	29.537	39.63	0.229	10.53	3800	2829	60.04	27.858	2.6491
69.94	29.502	39.51	0.233	10.58	3800	2840	59.80	27.819	2.6526
69.92	29.491	39.55	0.208	10.42	3800	2857	60.12	27.805	2.6279
69.92	29.494	39.60	0.196	10.59	3799	2891	59.82	27.845	2.6142
70.01	29.535	39.48	0.190	10.40	3800	2881	59.62	27.836	2.6064
70.01	29.532	39.55	0.199	10.48	3800	2886	59.58	27.850	2.6159
69.97	29.538	39.48	0.194	10.21	3800	2893	59.79	27.799	2.6112
69.95	29.519	39.52	0.181	10.31	3800	2895	59.45	27.857	2.5967
70.06	29.565	39.48	0.185	10.17	3800	2903	59.58	27.853	2.6003
69.98	29.522	39.40	0.188	9.90	3800	2892	59.59	27.782	2.6040
69.99	29.539	39.52	0.191	9.90	3800	2891	59.49	27.854	2.6076
70.00	29.552	39.55	0.188	9.52	3800	2894	59.33	28.071	2.6040
70.00	29.573	39.68	0.205	9.42	3799	2885	59.11	28.044	2.6226
70.13	29.652	39.73	0.209	9.28	3799	2870	59.16	27.999	2.6267
70.10	29.639	39.83	0.255	9.79	3799	2865	59.29	28.004	2.6735
70.08	29.641	39.91	0.259	9.51	3784	2860	59.26	27.956	2.6775
70.04	29.588	40.01	0.271	9.74	3765	2911	59.04	27.924	2.6884
70.03	29.573	40.01	0.259	9.83	3759	3795	58.65	28.002	2.6672
69.76	29.439	40.04	0.269	9.85	3763	3808	58.84	27.917	2.6781
69.95	29.539	39.91	0.260	9.99	3762	3799	58.62	28.108	2.6683
70.06	29.595	39.90	0.250	10.30	3768	3790	58.94	28.205	2.6585
70.02	29.575	39.87	0.261	10.37	3778	3785	59.02	28.260	2.6694
69.91	29.508	39.83	0.253	10.40	3791	3802	59.15	28.292	2.6619
70.09	29.617	39.80	0.236	10.42	3799	3807	59.30	28.258	2.6444
69.82	29.460	39.65	0.231	10.49	3822	3851	59.23	28.263	2.6403
70.09	29.610	39.68	0.234	10.41	3801	3842	59.02	28.215	2.6420
70.07	29.591	39.65	0.225	9.74	3799	3839	59.13	28.210	2.6332
70.16	29.654	39.65	0.242	9.94	3798	3814	59.36	28.235	2.6503
69.89	29.502	39.69	0.252	9.77	3800	3813	59.31	28.255	2.6613
69.99	29.548	39.67	0.226	9.92	3800	3808	59.74	28.298	2.6356
70.08	29.586	39.61	0.234	9.51	3801	3799	59.40	28.290	2.6426
70.04	29.561	39.68	0.262	10.09	3799	3800	59.62	28.296	2.6700
69.99	29.524	39.59	0.227	10.14	3799	3785	59.75	28.285	2.6368
70.03	29.554	39.61	0.189	10.35	3801	3780	59.78	28.300	2.5967
70.03	29.574	39.77	0.165	10.09	3796	3786	59.84	28.191	2.5698
69.75	29.410	39.74	0.148	9.67	3784	3802	60.89	26.484	2.5538
70.00	29.555	39.94	0.142	10.47	3748	3778	61.00	26.392	2.5450
70.07	29.513	40.05	0.152	10.65	3735	3772	60.82	26.314	2.5556
70.04	29.606	40.11	0.205	10.67	3732	3773	60.86	26.280	2.6171
70.03	29.596	40.19	0.238	10.30	3785	3768	61.41	26.225	2.6509

Generalization dataset

t	p	l	a	$f1$	$f2$	$f3$	$f4$	$f5$	MI
70.10	29.650	40.16	0.228	9.97	3804	3759	60.92	26.182	2.6403
69.94	29.547	40.19	0.244	10.17	3800	3759	60.83	26.204	2.6567
69.99	29.572	40.12	0.238	10.62	3800	3751	60.68	26.200	2.6503
70.12	29.633	40.04	0.200	10.72	3799	3729	61.06	26.224	2.6112
70.11	29.624	40.05	0.216	10.68	3801	3718	61.20	26.218	2.6279
69.86	29.485	40.09	0.199	10.95	3800	3737	61.06	26.205	2.6118
70.22	29.690	40.16	0.196	11.13	3799	3707	60.78	26.211	2.6064
69.93	29.536	40.17	0.186	11.11	3800	3713	60.91	26.247	2.5980
69.99	29.587	40.00	0.178	11.37	3800	3718	60.87	26.266	2.5882
70.07	29.613	40.06	0.216	11.46	3800	3701	60.49	26.252	2.6279
69.84	29.490	40.10	0.191	11.15	3800	3712	60.35	26.228	2.6027
70.10	29.623	40.29	0.203	11.03	3802	3708	60.35	26.276	2.6142
70.03	29.581	39.93	0.179	11.17	3801	3709	60.44	26.244	2.5876
70.06	29.655	39.72	0.228	11.44	3803	3778	60.33	26.233	2.6391
70.09	29.648	39.74	0.215	10.79	3798	3788	60.55	26.209	2.6255
69.87	29.494	39.91	0.223	11.00	3798	3788	60.59	26.188	2.6350
70.11	29.622	39.93	0.229	10.89	3800	3763	60.74	26.162	2.6403
70.02	29.567	39.91	0.237	11.15	3800	3752	60.79	26.134	2.6485
69.81	29.457	39.89	0.231	8.28	3802	3737	60.64	26.169	2.6438
70.05	29.596	39.91	0.239	3.18	3799	3712	60.50	26.203	2.6503
69.71	29.361	35.99	0.218	4.33	3811	3727	60.59	26.184	2.6238
69.90	29.507	32.91	0.214	4.87	3800	3723	61.95	26.219	2.6142
69.79	29.430	32.84	0.253	4.31	3802	3744	62.50	26.213	2.6532
69.83	29.454	32.86	0.191	4.49	3799	3713	59.35	26.123	2.5876
69.96	29.538	31.31	0.189	3.02	3800	3722	59.52	26.130	2.5821
70.00	29.562	31.49	0.218	3.35	3800	3752	59.56	26.150	2.6112
70.00	29.564	31.41	0.209	3.46	3800	3764	59.76	26.185	2.6027
70.02	29.576	31.47	0.217	3.26	3800	3763	60.44	26.193	2.6112
70.05	29.574	31.41	0.223	3.09	3800	3783	62.52	26.267	2.6189
69.98	29.541	31.37	0.221	3.32	3800	3794	62.50	26.256	2.6171
69.99	29.540	32.83	0.225	3.44	3794	3798	62.65	26.283	2.6243
70.00	29.562	33.57	0.208	3.32	3800	3795	62.88	26.268	2.6088
70.02	29.570	33.68	0.231	3.51	3797	3789	62.88	26.295	2.6320
70.06	29.593	35.27	0.226	2.97	3799	3753	62.76	26.254	2.6314
70.00	29.563	36.22	0.231	2.60	3799	3790	62.54	26.290	2.6368
70.03	29.577	36.29	0.216	2.94	3801	3783	62.50	26.285	2.6220
70.01	29.564	36.31	0.214	3.47	3799	3768	62.58	26.298	2.6202
70.05	29.591	36.26	0.220	2.81	3800	3777	62.70	26.339	2.6261
69.98	29.551	36.22	0.231	2.85	3800	3799	62.52	26.332	2.6368
70.01	29.582	36.20	0.226	3.18	3800	3810	62.94	26.357	2.6314
69.97	29.562	36.28	0.222	2.62	3800	3822	62.67	26.370	2.6285
70.03	29.602	36.24	0.221	2.07	3801	3832	62.77	26.410	2.6267
69.95	29.567	36.17	0.210	2.71	3799	3849	62.45	26.418	2.6165
69.99	29.588	36.16	0.203	6.25	3801	3848	62.28	26.404	2.6088
69.99	29.580	36.17	0.208	6.24	3799	3828	62.88	26.219	2.6136
70.02	29.594	36.31	0.208	6.14	3800	3813	62.74	26.178	2.6136
70.05	29.608	36.28	0.210	6.52	3800	3805	62.54	26.204	2.6154
70.03	29.574	36.33	0.215	6.68	3799	3781	62.42	26.392	2.6208
69.96	29.536	36.35	0.213	6.72	3799	3764	62.48	26.352	2.6195
70.02	29.569	36.44	0.224	6.82	3801	3754	62.16	26.337	2.6297

Manuscript received July 6, 2010, and revision received Apr. 2, 2011.