Huaqin Jiang
Yundu Xiao
Jiubao Li
Xinggao Liu

State Key Laboratory of Industrial Control Technology, Department of Control Science and Engineering, Zhejiang University, Hangzhou, China.

Research Article

Prediction of the Melt Index Based on the Relevance Vector Machine with Modified Particle Swarm Optimization

A novel chemical soft-sensor approach for the prediction of the melt index (MI) in the propylene polymerization industry is presented. The MI is considered as one of the important variables of quality that determine the product specifications. Thus, a reliable estimation of the MI is crucial in quality control. An accurate optimal predictive model of MI values with the relevance vector machine (RVM) is proposed, where the RVM is employed to build the MI prediction model; a modified particle swarm optimization (MPSO) algorithm is then introduced to optimize the parameter of the RVM, and the MPSO-RVM model is thereby developed. An online correcting strategy (OCS) is further carried out to update the modeling data and to revise the model's parameter self-adaptively whenever model mismatch happens. Based on the data from a real polypropylene production plant, a detailed comparison is carried out among the least squares support vector machine (LS-SVM), RVM, MPSO-RVM, and OCS-MPSO-RVM models. The research results reveal the prediction accuracy and validity of the proposed approach.

Keywords: Melt index prediction, Online correcting strategy, Particle swarm optimization, Propylene polymerization industry, Relevance vector machine

Received: August 16, 2011; *revised:* September 27, 2011; *accepted:* December 06, 2011

DOI: 10.1002/ceat.201100437

1 Introduction

Polymer production has a critical influence on some aspects of industry, military, economy and so on. The melt index (MI) of polypropylene (PP) determines the grade of the product as well as other product properties, so it is considered as one of the most important quality variables in the practical industrial polymerization process. However, the MI of PP is usually sampled online and then measured off-line with an analytical procedure in the laboratory, which is not only costly but also time consuming (2–4 h) [1]. Therefore, the development of an MI online estimation model is significant, not only as an online sensor but also as a forecasting system. Normally, an online analyzer can be constructed with the mechanism of polymerization [2–4]. However, it is often challenged by the engineering activity and the relatively high complexity of the kinetic behavior and operation of the polymer plants.

A number of researchers have attempted to find modeling approaches for the prediction of the MI, for application in the industrial process [5–9]. Embiruçu et al. [10], Lee et al. [11] and Tian et al. [12] developed models based on the process mass and energy balance. Because the processes are usually highly nonlinear, and because neural networks (NN) can approximate any continuous nonlinear function, NN and improved NN have been applied to system modeling and control in different fields [13–17]. Shi et al. [18] and Shi and Liu [19] have developed several soft-sensor models for MI prediction, based on ICA-MSA-RBF (independent component analysis, multi-scale analysis and radial basis function) NN and PCA-MS-RBF (principal component analysis, multi-scale analysis and radial basis function) NN. Zhang et al. [20] sequentially presented a bootstrap aggregated NN model that could overcome the problems of a single NN, such as over-fitting and the lack of generalization capability. To avoid the problem of NN, Han et al. [21] employed three approaches, which were supported vector machines (SVM), partial least squares (PLS) [22] and artificial NN, for the MI estimation of the styrene-acrylonitrile (SAN) and PP process. Park et al. [23] employed mechanical methods such as the PLS method and the support vector regression (SVR) method to predict the melt flow index (MFI) in the high-density polyethylene process. A detailed comparison among the standard SVM, the least squares-SVM (LS-SVM), and the weighted LS-SVM (WLS-SVM) models

Correspondence: Prof. X. Liu (liuxg@iipc.zju.edu.cn), State Key Laboratory of Industrial Control Technology, Department of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China.

was carried out by Shi and Liu [24]. Although these works have achieved a high level of MI prediction accuracy, greater performance in prediction and better universality of the estimation model are still the first-line goal in academia and the industrial community.

The relevance vector machine (RVM) is a sparse learning algorithm, similar to the SVM [25] and the LS-SVM in many respects, but capable of providing a fully probabilistic output. It allows avoiding the set of free parameters that the SVM has. However, up to now, little information on RVM applications in MI prediction of PP processes has been reported in the literature. In this paper, the RVM [26] is therefore explored to predict the MI according to a group of process variables in propylene polymerization that can be easily obtained.

The effectiveness of the RVM method largely depends on the value of the parameter. However, a general set of parameters cannot always ensure the RVM to achieve the desired performance in the highly complex and nonlinear PP process. Therefore, an optimization of the RVM parameter is essential to improve the performance of the RVM model in the MI prediction. Previous studies have shown lots of artificial intelligent algorithms in handling optimization problems, such as particle swarm optimization (PSO), ant colony optimization (ACO), genetic algorithms (GA) and so on [27]. Among these intelligent algorithms, PSO presents some interesting characteristics, like easy implementation procedure, high performance in different optimization problems, and high efficiency in convergence to desirable optima. Hence, the PSO is used to optimize the RVM parameter. To improve the basic PSO, a modified PSO (MPSO), by modifying the inertia weight, is proposed. Finally, the MPSO is used to optimize the RVM parameter, and the MPSO-RVM model is obtained.

Since the system is dynamic, it is necessary to modify the MPSO-RVM model when a new sample is added into the training dataset. An online correcting strategy (OCS) is therefore carried out to adjust the parameter online. Finally, the best OCS-MPSO-RVM model is achieved.

The rest of the paper is organized as follows: Sect. 2 provides the theoretic descriptions of the RVM, MPSO, and OCS. The procedural steps and flow chart of the OCS-MPSO-RVM model are presented in detail. The details of the experiment are given in Sect. 3 and the results of the experiments are presented and discussed in Sect. 4.

2 Theory

2.1 Relevance Vector Machine

The detailed description of the RVM is given in the following. Then, some advantages of the RVM algorithm compared with the SVM [28, 29] or the LS-SVM [30] are presented at the end of this section.

The RVM, pioneered by Tipping [26], is a sparse learning algorithm, similar to the SVM in many aspects, but capable of delivering a fully probabilistic output. It acquires relevance vectors and weights by maximizing a marginal likelihood. The products of weights and kernel functions give the structure of the RVM. A kernel function means a set of basic functions projecting the input data into a high-dimensional feature space.

Given a sample set of input-target pairs $\{x_n,t_n\}_{n=1}^{N}$, considering scalar-valued target functions only. The targets are from the model with additive noise:

$$t_n = y(x_n, w) + e \tag{1}$$

where e is the random noise, which is assumed to be independent zero-mean Gaussian distributed with variance σ^2. Like in the LS-SVM model, the function $y(x)$ is defined as follows:

$$y(x, w) = \sum_{i=1}^{N} w_i K(x, x_i) + w_0 = \sum_{i=1}^{N} w_i \Phi(x) \tag{2}$$

where $w = [w_0, w_1,...,w_N]$ is the weighted parameter vector of the basic function $\Phi(x)$; here, it is given as $\Phi(x) = [1,K(x,x_1), K(x,x_2),..., K(x,x_N)]^T$.

The targets can be given as $p(t_n|x_n) = N(t|y(x_n), \sigma^2)$. Due to the assumption of independence of t_n, the likelihood of the complete dataset can be written as :

$$p(t|w, \sigma^2) = \frac{1}{2\pi\sigma^2} \exp\{-\frac{1}{2\sigma^2} \parallel t - \Phi(x)w \parallel\} \tag{3}$$

where $t = (t_1,t_2,...t_N)$, $w = (w_0, w_1,...,w_N)$, and Φ is the $N \times (N+1)$ design matrix. In order to obtain optimal values of w and σ^2, the maximum likelihood estimation is expected. However, with as many parameters in the model as training data samples, the over-fitting problem can be caused. Here, RVM adopts a Bayesian perspective and constrains w and σ^2 by defining a prior probability distribution over the weights:

$$\begin{aligned} p(w|a) &= \prod_{i=1}^{N} N(w_i|0, a_i^{-1}) \\ &= \frac{1}{(2\pi)^{(N+1)/2}} \prod_{i=1}^{N} a_i^{1/2} \exp(-\frac{a_i w_i^2}{2}) \end{aligned} \tag{4}$$

$$p(a) = \prod_{i=1}^{N} \text{gamma}(a_i|a, b) \tag{5}$$

$$p(\beta) = \text{gamma}(\beta|a, b) \tag{6}$$

where $b = \sigma^{-2}$, a is an $N+1$ hyper-parameter, and gamma $(a|a, b)$ is defined as

$$\text{gamma}(a|a, b) = 1\Gamma(a)^{-1} b^a a^{a-1} e^{-ba} \Gamma(a) = \int_0^\infty t^{a-1} e^{-t} dt \tag{7}$$

The parameters a, b, c and d can be set as small values, such as $a = b = c = d = 10^{-5}$, so it can make the defined priors non-informative.

Having defined the prior probability of the parameter set, the posterior over weights can be calculated through the Bayesian rule:

$$\begin{aligned} p(w|t, a, \sigma^2) &= \frac{p(t|w, \sigma^2) p(w|a)}{p(t|a, \sigma^2)} \\ &= \frac{1}{(2\pi)^{(N+1)/2}} |\Sigma|^{-1/2} \exp\{-\frac{1}{2}(w-\mu)^T \Sigma^{-1} (w-\mu)\} \end{aligned} \tag{8}$$

where the posterior covariance and mean are defined as follows:

$$\Sigma = (\sigma^{-2} 2\Phi^{\mathrm{T}} 2\Phi + A)^{-1} \tag{9}$$

$$\mu = \sigma^{-2} \Sigma \Phi^{\mathrm{T}} t \tag{10}$$

where $A = \mathrm{diag}(a_1, a_2, ..., a_{\mathrm{N}})$. The likelihood distribution over the training targets can be marginalized by integrating the weights to obtain the marginal likelihood for the hyper-parameters; it is computable and given by Tipping [26]:

$$\begin{aligned} p(t|a, \sigma^2) &= \int p(t|w, \sigma^2) p(w|a) \mathrm{d}w \\ &= (2\pi)^{-\mathrm{N}/2} |C|^{-1/2} \exp\{\tfrac{1}{2} t^{\mathrm{T}} C^{-1} t\} \end{aligned} \tag{11}$$

where the covariance is given by $C = \sigma^2 I + \Phi A^{-1} \Phi^{\mathrm{T}}$.

Values of a and σ^2 cannot be obtained in closed form. The procedure is the following type-‖ maximum likelihood method. We use log values to find a maximum in Eq. (12):

$$L = \log p(t|\log a, \log \sigma^2) + \sum_{\mathrm{i}=0}^{\mathrm{N}} \log p\,(\log a_{\mathrm{i}}) + \log p(\log \sigma^2) \tag{12}$$

Obtain the first derivatives of this value with respect to a and σ^2, and set them to zero. An iterative re-estimation method is required [24]:

$$a_{\mathrm{i}}^{\mathrm{new}} = \frac{1 - a_{\mathrm{i}}\, \Sigma_{\mathrm{ii}}}{\mu_{\mathrm{i}}^2} \tag{13}$$

$$(\sigma^2)^{\mathrm{new}} = \frac{\| t - \Phi\mu \|^2}{N - \sum_{\mathrm{i}=1}^{\mathrm{N}} (1 - a_{\mathrm{i}} \Sigma_{\mathrm{ii}})} \tag{14}$$

The learning algorithm proceeds by repeating Eqs. (13) and (14), while updating the posterior statistics Σ and μ from Eqs. (9) and (10), until all a_{i} are smaller than the given value so that the corresponding model parameters w_{i} tend to zero.

Since the sparseness of the RVM is better than those of the SVM and the LS-SVM, which can lead to significant reduction in the computational complexity of the decision function, the computation efficiency of online soft-sensor prediction can be greatly improved by the RVM. Compared to the traditional SVM and LS-SVM methods, the RVM method can provide the probability distribution of the output value, which is very useful for statistical judgment and decision making.

In summary, the advantages of the RVM method can be listed as follows:

- The uncertainty of the output prediction value can be characterized.
- The RVM has better sparseness than the SVM and LS-SVM, which can reduce online prediction complexity.
- The RVM does not need to estimate the error/margin tradeoff parameter C, which can reduce the computational time.
- The kernel function does not need to satisfy the Mercer condition.

2.2 Modified Particle Swarm Optimization

The basic PSO algorithm is a method for optimizing hard numerical functions, originally developed by Kennedy and Eberhart [31], based on a social psychological metaphor of behaviors of flocks of birds and schools of fish. By randomly initializing a set of candidate solutions, the PSO successfully leads to a global optimum.

In PSO, each potential solution is represented as a particle, which is dynamically adjusted by updating the velocity of each particle, according to the experience of neighbor particles and its own experience. In detail, two properties associated with each particle are position x and velocity v. In each iteration, a fitness function is evaluated for all the particles and is used to determine the number of iterations. They are given as

$$x^{\mathrm{i}} = (x^{\mathrm{i}1}, x^{\mathrm{i}2}, \cdots, x^{\mathrm{iD}}) \tag{15}$$

$$v^{\mathrm{i}} = (v^{\mathrm{i}1}, v^{\mathrm{i}2}, \cdots v^{\mathrm{iD}}) \tag{16}$$

$$\mathrm{fitness} = (y_{\mathrm{i}}^* - \hat{y}_{\mathrm{i}})^2 / 2 \tag{17}$$

where $D = 1$ is size of the particle for the 1-dimension problem. $v^{\mathrm{i}} \in [-v_{\max}, v_{\max}]$ and $v_{\max}$ is the designed maximum velocity. y_{i}^* and $\hat{y}_{\mathrm{i}}$ denote the measured value of each iteration and the real value, respectively. The fitness indicates the error of prediction. So the lower the fitness, the better is the solution. The velocity of each particle is updated by keeping track of two best positions. One is the current best position of a particle, named p_{best}, the other is the best position of the whole population, called g_{best}. Hence, the velocity and position of a particle are updated as follows:

$$v_{\mathrm{k}+1}^{\mathrm{i}} = w_{\mathrm{k}} v_{\mathrm{k}}^{\mathrm{i}} + c_1 r_1 (p_{\mathrm{best}} - x_{\mathrm{k}}^{\mathrm{i}}) + c_2 r_2 (g_{\mathrm{best}} - x_{\mathrm{k}}^{\mathrm{i}}) \tag{18}$$

$$x_{\mathrm{k}+1}^{\mathrm{i}} = x_{\mathrm{k}}^{\mathrm{i}} + v_{\mathrm{k}+1}^{\mathrm{i}} \tag{19}$$

where w_{k} is the inertia weight, controlling the impact of the previous velocity on its current one. c_1 and c_2 are positive constants, called acceleration coefficients. r_1 and r_2 are stochastic coefficients that are uniformly distributed in the interval [0,1].

In order to modify w_{k}, the method given below is used:

$$w_{\mathrm{k}} = w_{\max} - (w_{\max} - w_{\min}) \times (k - 1) / \mathrm{iter}_{\max} \tag{20}$$

In the MPSO, k is the current number of iteration, and $\mathrm{iter}_{\max}$ is the maximal number of iterations; the inertia weight w_{k} starts with a high value $w_{\max}$ and nonlinearly decreases to $w_{\min}$.

According to Eqs. (18) and (19), the population of particles tends to cluster together from different directions. The PSO algorithm runs until the termination criterion is satisfied. Fig. 1 represents the convergence property of the MPSO algorithm in the process of searching for the minimum error of prediction. After 100 iterations, fitness is close to the minimum point. It is clearly obvious that the method can obtain good computation efficiency and convergence property. Fig. 2 shows that the inertia weight w_{k} of the MPSO nonlinearly decreases from $w_{\max}$ to $w_{\min}$ with increasing iterations.

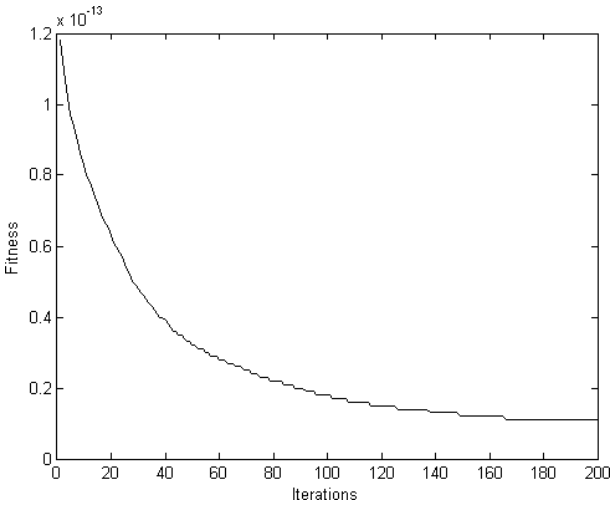

Figure 1. Convergence behavior of the MPSO algorithm.

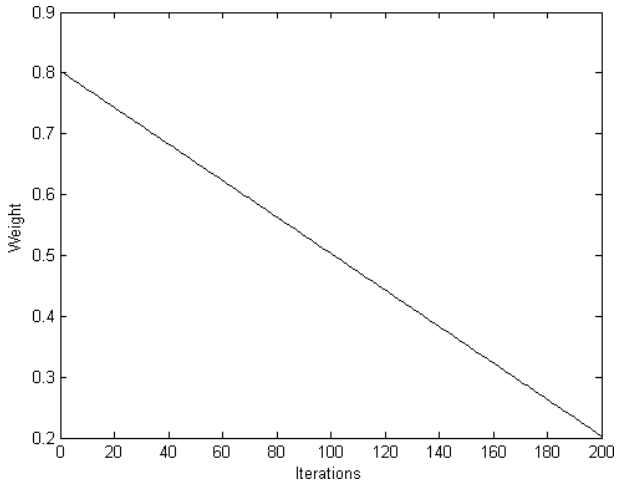

Figure 2. Performance of the inertia weight of MPSO.

2.3 Hybrid MPSO-RVM Algorithm

Based on Sect. 2.1 and 2.2, a modified PSO optimizes the kernel parameter σ of the RVM, and then the MPSO-RVM model is developed. The σ value is strongly related to the precise prediction of the model. The flow chart of the MPSO-RVM is shown in Fig. 3. Furthermore, the detailed procedure of the MPSO-RVM model is outlined as follows:

Step 1: Create the initial position of particles x with one dimension for the parameter of the RVM. Initialize the velocity of particles v, the maximum number of iteration $\text{iter}_{\max}$, and the number of particles *Popsize*.

Step 2: According to the model RVM, the initial result $\hat{y}_i$ is obtained. Evaluate the initial fitness of each particle according to Eq. (17). Then, let the particles with the best fitness be the initial p_{best}. The initial g_{best} is obtained from the p_{best} particles.

Step 3: According to Eq. (20), calculate the weights w_k.

Step 4: Update the position and the velocity of every particle x^i according to Eqs. (18) and (19). In this paper, the parameters are taken as $c_1 = c_2 = 2.0$. If $v^i > v_{\max}$, refine $v^i = v_{\max}$; if $v^i < -v_{\max}$, refine $v^i = -v_{\max}$.

Step 6: Set $i = 1$, with the updated x^i, calculate the fitness again.

Step 7: Update the p_{best} according to the fitness; if the fitness of the current particle is lower than the fitness of the initial p_{best}, use the current p_{best} and update the g_{best}.

Step 8: $i = i + 1$, if $i > Popsize$, go to Step 9; else go back to Step 6.

Step 9: Taking the updated p_{best} as the parameter of the model RVM, the new prediction can be obtained. Then update the p_{best} and the g_{best} according to the new fitness.

Step 10: $k = k + 1$, if $k > \text{iter}_{\max}$, go to Step 11; else go to Step 3.

Step 11: Take the updated g_{best} as the optimized parameter of the model RVM. The final solution of the optimizing problem is found.

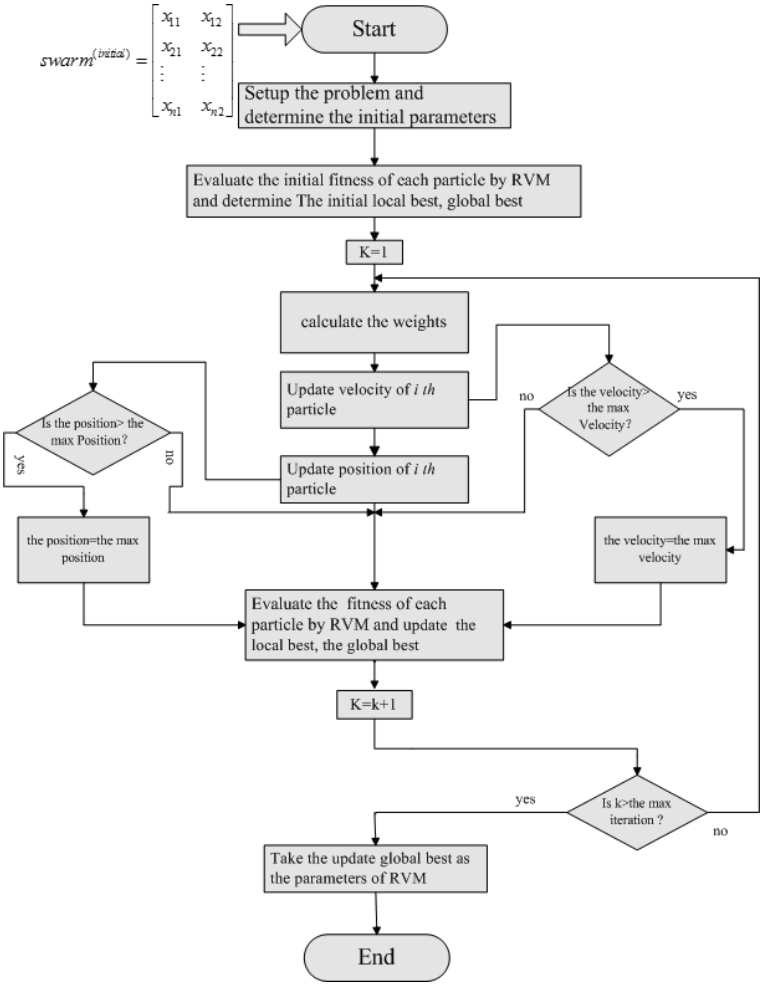

Figure 3. Flow chart of the hybrid MPSO-RVM algorithm.

2.4 Online Correcting Strategy

The static MPSO-RVM model with fixed parameters does not take into account the problem how to modify the model whenever a new sample is available, so it cannot be adapted to a dynamic system. With new samples gradually added to the training dataset, a bigger prediction error will be generated. It is difficult for the model to be used in an online MI prediction system. Therefore, it is quite important to modify the MPSO-RVM model when a new sample is added into the training dataset.

In order to obtain higher prediction precision, the OCS is used. The main idea of the OCS is to dynamically modify the training dataset with new samples and then find the best parameter for the RVM model step by step. The OCS can quickly find a new MPSO-RVM model that works well both on the history dataset and the new added data. The flow chart of the OCS is shown in Fig. 4. The analytic value of the 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 and the input variables x at time t, is then put into the OCS. The OCS first calculates the difference between mi_a and mi_p. For example, if $|mi_a - mi_p| >$ MXAE, where MXAE is the mean absolute error of the MPSO-RVM model performs on the training dataset, new training data $\{x(t), mi_a(t)\}$ should be added into the training dataset. Then the MPSO algorithm starts using the new training dataset to get a new MPSO-RVM model.

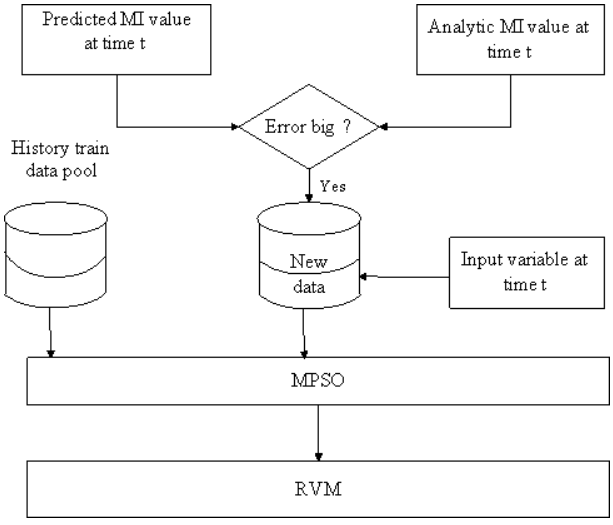

Figure 4. Flow chart of the online correcting strategy.

3 Experimental Case Study

A simple schematic diagram of a propylene polymerization process, which has been recently operated for commerce in a real plant, is shown in Fig. 5. There is a chain of reactors in series, two continuous stirred-tank reactors (CSTR) and two fluidized-bed reactors (FBR). The polymerization reaction begins with a liquid phase in the two CSTR as first two reactors and is completed in a vapor phase in the two FBR as the third and fourth reactors to produce the powdered polymer product. The MI of the PP, which depends on the reactant composition, the reactor temperature, etc., can determine the properties and quality of the product.

3.1 Selection of Input Variables

To develop a prediction model to estimate the MI, a group of easily measured variables, which are acquired from the historical logs recorded in a real propylene polymerization plant, were chosen. Totally, nine process variables (T, p, l, a, $f1$, $f2$, $f3$, $f4$, $f5$) are chosen as the input data according to the workers' experience and our mechanism analysis [19, 24]. T, p, l, a: Process temperature, pressure, level of liquid, percentage of hydrogen in the vapor phase; $f1$, $f2$, $f3$: flow rates of three streams of propylene; $f4$, $f5$: flow rates of the catalyst and the aid-catalyst, respectively. The data are filtered to discard abnormal situations and to improve the quality of the prediction results. The variables are normalized with the method of statistical normalization. The detailed algorithm is as follows: from the data as a vector, the mean of the data is subtracted, and this difference is divided by the standard deviation; finally, normal data will be obtained with mean zero and variance = 1. The normal data

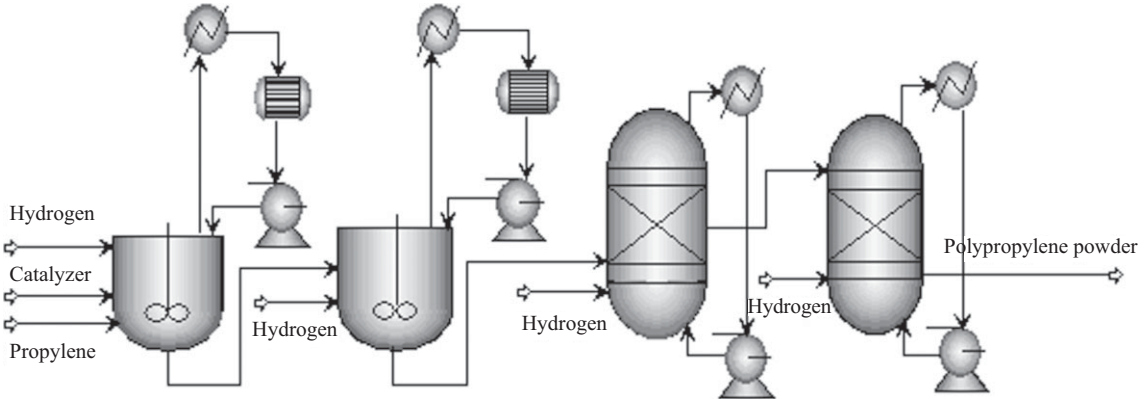

Figure 5. General scheme of propylene polymerization.

from the time records are separated into training, testing, and generalization datasets.

A collection of 85 pairs of input-output data are used in this research. For the LS-SVM, RVM, and MPSO-RVM models, 40 pairs are used as the training dataset; the others are used as the test set. For the OCS-MPSO-RVM model, 20 pairs are used as the training dataset; the remaining 65 pairs are used as the testing dataset and the generalization dataset, where the number of the testing dataset is 40 and the generalization dataset is 25. It is noted that the training set is taken from the same batch as the testing dataset, while the generalization dataset is obtained from a different one. So, in this paper, the performance of the test can reflect the model's prediction accuracy and the performance of generalization can reflect the model's universality.

3.2 Performance Criterion

In this paper, the difference between the output of the models and the real output is considered as the error and represented in several ways, including mean absolute error (MAE), maximum mean absolute error (MXAE), mean relative error (MRE), root mean square error (RMSE), and Theil's inequality coefficient (TIC) [32]. They are defined as follows:

$$\mathrm{MAE} = \frac{1}{n}\sum_{\mathrm{i}=1}^{\mathrm{n}} \left|y_{\mathrm{i}} - \hat{y}_{\mathrm{i}}\right| \tag{21}$$

$$\mathrm{MXAE} = \max\left|y_{\mathrm{i}} - \hat{y}_{\mathrm{i}}\right| \tag{22}$$

$$\mathrm{MRE} = \frac{1}{n}\sum_{\mathrm{i}=1}^{\mathrm{n}} \frac{|y_{\mathrm{i}} - \hat{y}_{\mathrm{i}}|}{y_{\mathrm{i}}} \times 100\% \tag{23}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{\mathrm{i}=1}^{\mathrm{n}} (y_{\mathrm{i}} - \hat{y}_{\mathrm{i}})^2} \tag{24}$$

$$\mathrm{TIC} = \frac{\sqrt{\frac{1}{n}\sum_{\mathrm{i}=1}^{\mathrm{n}} (y_{\mathrm{i}} - \hat{y}_{\mathrm{i}})^2}}{\sqrt{\sum_{\mathrm{i}=1}^{\mathrm{n}} y_{\mathrm{i}}^2} + \sqrt{\sum_{\mathrm{i}=1}^{\mathrm{n}} \hat{y}_{\mathrm{i}}^2}} \tag{25}$$

where $e_{\mathrm{i}} = y_{\mathrm{i}} - \hat{y}_{\mathrm{i}}$, $\bar{e} = \frac{1}{n}\sum_{\mathrm{i}=1}^{n} e$, and y_{i} and $\hat{y}_{\mathrm{i}}$ denote the measured value and predicted result, respectively.

In order to assess the performance of the proposed OCS-MPSO-RVM model, the other three models, i.e. LS-SVM, RVM, MPSO-RVM, are also developed as comparison basis based on the group's previous work [24, 33, 34]. MAE, MXAE, MRE, and RMSE confirm the prediction accuracy of the proposed methods. The standard deviation (STD) indicates the predictive stability of the methods, and the TIC indicates the level of agreement between the proposed models and the real process.

3.3 Algorithm Configuration

In the proposed OCS-MPSO-RVM method, many parameters need to be set carefully. In the MPSO algorithm, the particle number *Popsize* is set to 50, and c_1 and c_2 are initialized to 2.0. The initial value of the velocity vector is constrained into [–1,1], and the modified $v_{\max}$ is set to 2. For the inertia weight w_{k}, we set $w_{\max} = 0.8$, $w_{\min} = 0.2$. The algorithm stops when the $\mathrm{iter}_{\max}$ meets 200.

To obtain a good performance of the RVM model, the parameter is set differently in each operation process. At last, the one much better than the mean value is chosen in this paper.

4 Results and Discussion

Tab. 1 shows that the RVM model with MPSO has the best performance on the testing dataset overall. In a detailed study through error indicators mentioned in Section 3.2, the MPSO-RVM gives an MAE of 0.0055, compared with 0.0149 and 0.0842 obtained from the corresponding RVM and LS-SVM [24], respectively. In terms of MRE, the MPSO-RVM prediction accuracy is 0.21 % and that of the RVM is 0.60 %, much better than the LS-SVM accuracy (3.66 %) obtained by Shi and Liu [24]. Similar results are observed in terms of RMSE, with a decrease from 0.0800 to 0.0700. It is noted that the TIC of the MPSO-RVM (0.0040) is quite acceptable when compared with that of the RVM (0.0045) and the LS-SVM (0.0240), which indicates a good level of agreement between the proposed model and the real process. Moreover, the MXAE obtained by the MPSO-RVM model is 0.0900, which is close to that of the RVM (0.0800). So the RVM model has a better performance than the LS-SVM model obtained by Shi and Liu [24], and the RVM model optimized by the MPSO algorithm is the best of the three models. All in all, the MAE, MRE, RMSE, MXAE and TIC of the OCS-MPSO-RVM model are the smallest shown in Tab. 2, with decreased percentages of 24, 14, 94, 95, and 78 % compared to those of the MPSO-RVM, which is the best model shown in Tab. 1. Moreover, on the generalization data-

Table 1. Performance comparison of different soft sensors on the testing dataset.

Methods	MAE	MRE	RMSE	MXAE	TIC
LS-SVM [24]	0.0842	3.66 %	–	–	0.0240
RVM	0.0149	0.60 %	0.0800	0.0800	0.0045
MPSO-RVM	0.0055	0.21 %	0.0700	0.0900	0.0040

Table 2. Performance of the OCS-MPSO-RVM model on the test and generalization datasets.

OCS-MPSO-RVM	MAE	MRE	RMSE	MXAE	TIC
Test	0.0042	0.18 %	0.0041	0.0049	0.00088
Generalization	0.0039	0.15 %	0.0040	0.0040	0.00077

set, the performance of the OCS-MPSO-RVM model is also good, according to the results of MAE (0.0039), MRE (0.15 %), RMSE (0.0040), MXAE (0.0040), and TIC (0.00077). Therefore, the proposed OCS-MPSO-RVM model in this paper has a good universality.

A more distinct illustration of how much better the MPSO-RVM model works than the other two models on the testing dataset is shown in Fig. 6. The *x*-axis is the real MI value of the real plant, while the *y*-axis is the predicted value. The curves marked with circles, triangles and crosses are the prediction values by the LS-SVM, RVM and MPSO-RVM, respectively. Obviously, the marks that are close to the diagonal indicate that the predicted value approximates the real value. As visible, the OCS-MPSO-RVM model yields consistently good prediction and a performance that is a little better than that of the RVM and LS-SVM models.

Fig. 7 illustrates the performance of the OCS-MPSO-RVM model on the testing and generalization datasets. The model precisely predicts most of the testing and generalization datasets due to its online adjustment mechanism. Clearly, the OCS-MPSO-RVM model gives a near-reality MI value prediction, much more accurate than the single RVM model and the LS-SVM. Thus, it is proven that the OCS-MPSO-RVM model holds excellent performance in MI prediction, both statistically and graphically.

Tab. 3 compares the OCS-MPSO-RVM model with other models presented in the open literature [24, 33–36]; note that the industrial data of Refs. [24, 34] is the same as in this paper, while the data of Refs. [33, 35, 36] is different from this paper. So the results of Refs. [33, 35, 36] are for reference only. With the same industrial data, our work improves the prediction precision from MRE 3.27 % presented in [24] to MRE 0.15 %, and the RMSE of our work is the smallest, which indicates that the method presented in this article has obviously improved the prediction precision.

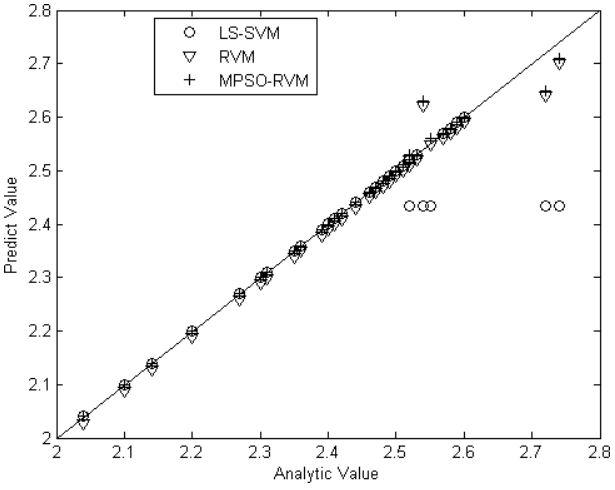

Figure 6. Comparative performance of the LS-SVM, RVM, MPSO-RVM models.

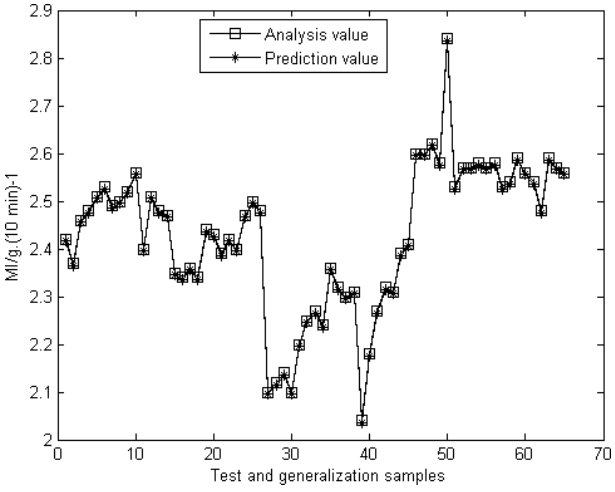

Figure 7. Performance of the OCS-MPSO-RVM model on the testing and generalization datasets.

Table 3. Comparison between the current work and earlier published articles.

Literature	Methods	MAE	MRE	RMSE	TIC
Cao [33]	Adaptive RBF	0.10	–	0.62	–
Shi and Liu [24]	WLS-SVM	0.0754	3.27 %	–	0.0223
Lou and Liu [34]	PCA-GA-RBF	0.022	0.84 %	0.0597	–
Kim and Yeo [35]	Mechanism model	–	–	0.46	–
Yeo et al. [36]	New RPLS	–	–	0.1466	–
This paper	OCS-MPSO-RVM	0.0042	0.15 %	0.0040	0.00077

5 Conclusions

The prediction of the MI of PP in a real industrial plant using the OCS-MPSO-RVM model, which aimed at continuous optimizing problems in MI prediction, is presented in this paper. With the many advantages of the RVM, the performance of the RVM is much better compared to the LS-SVM. The experimental results show that the RVM with the MPSO algorithm is more efficient than the pure RVM method. The OCS is applied in this research to modify the MPSO-RVM model, and by combining MPSO and OCS, we obtain a satisfactory MI prediction system. For comparison purposes, the LS-SVM, RVM, MPSO-RVM, OCS-MPSO-RVM are developed and evaluated. The estimation errors can be reduced by using the MPSO-RVM model, and can be further reduced by using the OCS-MPSO-RVM model. The application of the proposed training method to the test and generalization data obtained from an industrial polymerization plant demonstrated its effectiveness and reliability. The OCS-MPSO-RVM model predicts the MI with an MRE of 0.18 % on the test dataset, which is much more accurate than with the RVM model with an MRE of 0.60 %, and much better than with the LS-SVM model with an MRE of 1.57 %. The proposed modeling approach is therefore supposed to have a promising potential for practical use.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (Grant 50876093), the National High Technology Research and Development Program (863, Grant 2006AA05Z226), the Zhejiang Provincial Natural Science Foundation for Distinguished Young Scientists (Grant R4100133) and the International Cooperation and Exchange Project of Science and Technology Department of Zhejiang Province (Grant 2009C34008), and their supports are hereby acknowledged.

The authors have declared no conflict of interest.

References

[1] S. S. Bafna, A. M. Beall, *J. Appl. Polym. Sci.* **1997**, *65*, 277.
[2] K. B. McAuley, J. F. MacGregor, *AIChE J.* **1991**, *37*, 825.
[3] P. Sarkar, S. K. Gupta, *Polym. Eng. Sci.* **1993**, *33*, 368.
[4] T. F. McKenna, J. B. P. Soares, *Chem. Eng. Sci.* **2001**, *56*, 3931.
[5] F. Ahmed, S. Nazir, Y. K. Yeo, *Chem. Eng. J.* **2009**, *26*, 14.
[6] Y. Chen, X. G. Liu, *Polymer* **2005**, *46*, 9434.
[7] E. H. Lee, T. Y. Kim, Y. K. Yeo, *J. Chem. Eng. Jpn.* **2007**, *40*, 840.
[8] D. P. Shi, Z. H. Luo, A. Y. Guo, *Ind. Eng. Chem. Res.* **2010**, *49*, 4070.
[9] L. Xie, Y. J. Liu, H. Z. Yang, F. Ding, *IET Control Theory A.* **2010**, *4*, 784.
[10] M. Embiruçu, E. L. Lima, J. C. Pinto, *J. Appl. Polym. Sci.* **2000**, *77*, 1574.
[11] E. H. Lee, T. Y. Kim, Y. K. Yeo, *J. Chem. Eng. Jpn.* **2007**, *40*, 840.
[12] C. Q. Tian, Y. F. Liao, X. T. Li, *Int. J. Refrig.* **2006**, *29*, 270.
[13] Y. Li, S. H. Yang, Z. Zhang, Q. G. Wang, *IET Control Theory A.* **2010**, *4*, 197.
[14] Y. Zhang, H. Li, A. Hou, J. Havel, *Talanta* **2005**, *65*, 118.
[15] C. R. Zhang, Y. Z. Zhang, B. D. Zheng, *J. Comput. Appl. Math.* **2009**, *229*, 264.
[16] Z. H. Xiong, Y. X. Xu, J. Zhang, J. Dong, *Neural Comput. Appl.* **2008**, *17*, 425.
[17] B. Yu, Y. Shi, J. Huang, *J. Dyn. Syst. Meas. Control Trans. ASME* **2011**, *133*, 3.
[18] J. Shi, X. G. Liu, Y. X. Sun, *Neurocomputing* **2006**, *70*, 280.
[19] J. Shi, X. G. Liu, *Chin. J. Chem. Eng.* **2005**, *13*, 849.
[20] J. Zhang, Q. B. Jin, Y. M. Xu, *Chem. Eng. Technol.* **2006**, *29*, 442.
[21] I. S. Han, C. Han, C. B. Chung, *J. Appl. Polym. Sci.* **2005**, *95*, 967.
[22] G. Baffi, J. Morris, E. Martin, *Chem. Eng. Res. Des.* **2002**, *80*, 75.
[23] T. C. Park, T. Y. Kim, Y. K. Yeo, *Korean J. Chem. Eng.* **2010**, *27*, 1662.
[24] J. Shi, X. G. Liu, *J. Appl. Polym. Sci.* **2006**, *101*, 285.
[25] J. Taboada, J. M. Matías, C. Ordóñez, P. J. García, *J. Comput. Appl. Math.* **2007**, *204*, 84.
[26] M. E. Tipping, *J. Mach. Learn. Res.* **2001**, *1*, 211.
[27] W. F. Abd-El-Waheda, A. A. Mousab, M. A. El-Shorbagy, *J. Comput. Appl. Math.* **2011**, *235*, 1446.
[28] V. N. Vapnik, The Nature of Statistical Learning Theory, Springer-Verlag, New York **1995**.
[29] V. N. Vapnik, Statistical Learning Theory, John Wiley, New York **1998**.
[30] J. A. K. Suykens, J. Vandewalle, *Neural Process. Lett.* **1999**, *9*, 293.
[31] J. Kennedy, R. Eberhart, Particle swarm optimization, in *Proc. IEEE Int. Conf. Neural Networks*, IEEE Press, Piscataway **1995**.
[32] D. J. Murray-Smith, *Math. Comput. Model. Dyn.* **1998**, *4*, 5.
[33] J. Cao, G. Z. Wang, B. W. Xu, *J. Control. Decis.* **1999**, *14*, 339.
[34] W. Lou, X. G. Liu, *J. Petrochem. Univ.* **2007**, *20*, 82.
[35] T. Y. Kim, Y. K. Yeo, *Korean J. Chem. Eng.* **2010**, *27*, 1669.
[36] Y. K. Yeo, S. Nazir, F. Ahmed, *Chem. Prod. Process. M.* **2009**, *4*, 33.

