Online Optimization of Heated-Oil Pipeline Operation Based on Neural Network System Identification

Yu Zhang, Ph.D.[1]; Bo Wang[2]; and Xinjing Huang, Ph.D.[3]

Abstract: Traditional optimization methods of heated oil pipeline operation are usually offline or nondynamic, where steady-state models or complex and inaccurate physical models are utilized. To achieve online optimization with high adaptability and timeliness, this paper deploys a neural network (NN) to identify and imitate a real thermal and pressure response system instead of solving complicated model equations. The NN is trained and built by pipeline history data obtained from the Supervisory Control and Data Acquisition (SCADA) system. For a certain pipeline, real-time operating parameters like pressure and temperature are predicted by the NN once the inputs are given. Three different neural networks, a backpropagation neural network (BPNN), radial basis function neural network (RBFNN), and general regression neural network (GRNN), are compared to demonstrate their error-control abilities when predicting those parameters. Then the parameters are sent to particle swarm optimization and differential evolution (PSO-DE) optimization units to calculate the costs. The NN identification and PSO-DE optimization are applied to the Rizhao–Yizheng digital long crude oil pipeline. Experiments demonstrate that the three NNs have similar prediction accuracies, and the RBFNN and GRNN have better prediction stability than BPNN. Most of the prediction errors of the flow, outlet pressure, and inlet temperature can be controlled below 50 m^3/h, 0.1 MPa, and 0.1°C, respectively, and the online optimization is achievable. The maximum reduction of the total energy cost is 10.75%, and the saving effect is significant. **DOI: 10.1061/(ASCE)PS.1949-1204.0000421.**

Author keywords: Heated oil pipeline; Neural network; System identification; Online optimization; Algorithm.

Introduction

Pipelines are one of the most significant ways to transport crude oil. However, some crude oil, with a high freezing point, high content of wax, and high viscosity, is usually heated and pumped when transported in pipelines. Therefore, extra power and fuels are consumed during transport, which takes up 1%–3% of the total energy costs. To reduce costs in transportation, optimization of a heated pipeline operation is essential, which can be solved by seeking the best combinations and selections of heating and pumping stations within the safety requirements. In general long heated oil pipelines (HOPs) work continuously, which means, once the transport task is altered, stations should adjust their operating states to meet the demands of the new task, thereby allowing for online optimization.

Studies on HOP optimization started in the 1960s, when Yong and Jefferson (1961) used a dynamic programming method to determine the best pressure to use at each station. Gopal (1980) used a linear programming method to choose the best combination of pump stations. Kurak (1989) used power optimization software on an oil product pipeline and reduced the power costs of pump stations. However, these methods are oversimplified and mainly focus on steady-state models, which means offline and lacking dynamic adaptability. Then some intelligent algorithms were proposed to solve complex models. Hence, Simpson et al. (1994) presented a methodology for optimizing pipe networks using genetic algorithms (GAs) and compared the results with nonlinear programming and found the global optimum in relatively few evaluations. Ilich and Simonovic (1998) proposed an evolutionary algorithm to minimize the total cost of pumping in a liquid pipeline; their algorithm was an improvement in terms of optimality compared to the widespread gradient search methods. Botros et al. (2004) developed multiobjective function and safety constraints based on GAs. Gao et al. (2004) used a GA combined with a simulated annealing (SA) algorithm to solve an optimization model and successfully reduced total costs. Li et al. (2007) adopted a mixed particle swarm optimization (PSO) algorithm with mixed discrete random programming and applied it to the model, and the result performed better than just a PSO algorithm. Li et al. (2011) improved the pattern search algorithm integrated with the particle swarm method in gas transmission networks, and the simulation result was superior to the results of a simplex algorithm and the classic linear programming algorithm. Oliveira et al. (2016) proposed a framework that comprises a stochastic optimization model based on mixed-integer linear programming for scheduling a crude oil pipeline and a method for representing oil supply uncertainty. Zhang et al. (2017) used an improved PSO and Monte Carlo method to design subsea oil pipeline module, which can be applied to analyzing the thermohydraulic and structural force and solving the main pipeline parameters. Zhang et al. (2018) proposed a self-learning algorithm for pipeline scheduling, which can improve calculation speed and efficiency by itself.

The optimization algorithms for the three largest infrastructures—smart grids (Behera et al. 2015; Twaha and Ramli 2018; Abdi et al. 2017; Yan and Peng 2016), water pipe networks (Zhao et al. 2016; Makaremi et al. 2017), and heated oil pipelines (Lervik et al. 2018;

[1]Associate Professor, State Key Laboratory of Precision Measuring Technology and Instruments, Tianjin Univ., Tianjin 300072, China. Email: zhangyu@tju.edu.cn

[2]Graduate Student, State Key Laboratory of Precision Measuring Technology and Instruments, Tianjin Univ., Tianjin 300072, China. Email: 1226116797@qq.com

[3]Assistant Professor, State Key Laboratory of Precision Measuring Technology and Instruments, Tianjin Univ., Tianjin 300072, China (corresponding author). Email: huangxinjing@tju.edu.cn

Note. This manuscript was submitted on March 15, 2018; approved on April 22, 2019; published online on September 24, 2019. Discussion period open until February 24, 2020; separate discussions must be submitted for individual papers. This paper is part of the ***Journal of Pipeline Systems Engineering and Practice***, © ASCE, ISSN 1949-1190.

Pavlov et al. 2017; Wang et al. 2017; Zuo et al. 2018)—are similar, including mathematical and metaheuristic methods. Mathematical methods include network flow theory, linear programming, nonlinear programming, integer programming, and dynamic programming. Metaheuristic methods include PSO, GAs, Monte Carlo method, and SA. Faster algorithms generated in one domain can be easily transplanted to another domain. For the HOP operation optimization, the physics coupling needs updating in real time when potential solutions are being generated, which is the biggest difference compared to smart grids and water pipe networks. The ability to accurately predict each intermediate physical variable of HOP in real time is the key to whether optimization is rapid and successful.

Most of the aforementioned studies emphasize optimization modeling and solving. When judging whether or not some intermediate variables satisfy constraint conditions, such as inlet/outlet pressure and temperature, HOP physics models of pressure and thermal responses are used. However, some assumptions neglect too many details in predicting variables, and the models are so ideal that the prediction results are incorrect. Otherwise, the real-time computation of accurate models will be complicated and not suitable for online prediction. All in all, a long-distance HOP is a complex system with high nonlinear and multiple variables. A model without considering the nonlinearity or with too complicated algorithms is not appropriate for online optimization. Impractical operating processes can lead to false optimization results.

In this paper, artificial neural networks (NNs) are used to identify and imitate practical thermal and pressure response systems instead of solving complex models. With the input given by pipeline history data, the NNs calculate the corresponding output and send the identified results to PSO and differential evolution (PSO-DE) calculation units (Viani et al. 2017), and the optimization calculation also uses the network (Zhou et al. 2015). The PSO-DE algorithm was introduced in a previous publication (Zhou et al. 2015). There are two advantages to using system identification based on NNs: (1) because the actual input and output data are utilized to approximate the HOP system without assumptions, the NN model is very close to the actual system; (2) there is no finite-element simulation iteration, so the optimization is near real time, and training can be carried out both offline and online.

Optimization Goal

The goal of HOP optimization is to minimize the costs S in transportation, which consists of power costs S_P of electrical pumps and fuel costs S_F of heating devices. The objective function is given as Eq. (1) (Zhou et al. 2015):

$$F(\boldsymbol{X}) = S = S_P + S_F = \sum_{i=1}^{n}\sum_{j=1}^{m_i}\gamma_{ij}H_{p_{ij}}\frac{e_y K_1}{\eta_{p_{ij}}\eta_{e_{ij}}} + \sum_{i=1}^{n} e_x K_2 C\frac{(T_{in_i} - T_{out_i})}{\eta_{r_i} H_r} \quad (1)$$

In Eq. (1), power costs at the ith ($i = 1, 2, \ldots, n$) pumping station S_{Pi} are formed by costs $\sum_{j=1}^{m_i}\gamma_{ij}H_{p_{ij}}(e_y K_1/\eta_{p_{ij}}\eta_{e_{ij}})$ of jth ($j = 1, 2, \ldots, m$) pumps. For a certain pump, γ describes the on(1)/off(0) state, H_p is the delivery of a pump head, which can be tested by the difference between outlet pressure P_{out} and inlet pressure P_{in}, $\eta_{p_{ij}}$ is the efficiency of a pump, e_y is the electrical power price, $\eta_{e_{ij}}$ describes a pump's efficiency at converting power to pressure, K is a coefficient to dimenasions. Fuel costs at the ith heating station (settled with pumps at the same station) S_{Fi} are formed by the cost $e_x K_2 C[(T_{in_i} - T_{out_i})/\eta_{r_i} H_r]$ of the heating device, where T_{in} is the inlet temperature and T_{out} is the outlet temperature of the station, e_x is the fuel price, C is the specific heat capacity, H_r is the combustion value of fuels, and η_r describes the heating conversion efficiency of fuel to thermal energy.

Accordingly, the parameters of a pipeline model that the PSO-DE optimization requires are flow, pressure, and temperature. However, to identify the pipeline model and to efficiently use PSO-DE, the input and output variables of a NN need to be selected properly. In general the output variables are able to describe the objectives, and the input variables should have sensitive influence on the output variables. Usually there is more than one pump at stations that are settled along the transporting direction of a pipeline section. Based on practical engineering experiments, the input and output combination is assigned as follows: on-off states of pump groups $\boldsymbol{\gamma} = \{\gamma_{ij}\}_{n\times m}$, outlet temperature of all stations $\boldsymbol{T}_{out} = \{T_{out_i}\}_{n\times 1}$ combined as input variables; flux $\boldsymbol{Q}$, inlet temperature of all stations $\boldsymbol{T}_{in} = \{T_{in_i}\}_{n\times 1}$, inlet pressure of pumps at all stations $\boldsymbol{P}_{in} = \{P_{in_i}\}_{n\times 1}$, and outlet pressure of pumps at all stations $\boldsymbol{P}_{out} = \{P_{out_i}\}_{n\times 1}$ combined as output variables. Other parameters such as pressure drop at the first station and component variables of oil are constant in one specific transporting task.

Thus, the objective function Eq. (1) can be described as $F(\boldsymbol{X})$, and the independent variables matrix $\boldsymbol{X}$ and dependent variables matrix $\boldsymbol{O} = F(\boldsymbol{X})$ can be described as Eqs. (2)–(6):

$$\boldsymbol{X} = (\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_m, \boldsymbol{T}_{out})^{\mathbf{T}} \quad (2)$$

$$p_{out_i} = f_{pout_i}(\gamma_i, T_{in_i}) \quad (3)$$

$$p_{in_{i+1}} = f_{pin_i}(p_{out_i}, T_{out_i}) \quad (4)$$

$$T_{in_{i+1}} = f_{Tin_i}(p_{in_i}, T_{out_i}) \quad (5)$$

$$\boldsymbol{O} = (\boldsymbol{Q}, \boldsymbol{T}_{in}, \boldsymbol{P}_{in}, \boldsymbol{P}_{out})^{\mathbf{T}} \quad (6)$$

At the ith station, the outlet pressure p_{out_i} is determined by the pump on-off states γ_i and input temperature T_{in_i}. In the $(i+1)$th station, the inlet pressure $p_{in_{i+1}}$ is determined by the outlet pressure p_{out_i} and the outlet temperature T_{out_i} of the preceding station. At the $(i+1)$th station, the inlet temperature $T_{in_{i+1}}$ is determined by the inlet pressure p_{in_i} and the outlet temperature T_{out_i} of the ith station.

The constraint conditions, such as inlet/outlet pressure and temperature and maximum internal pressure, can be found in the authors' previous publication (Zhou et al. 2015). A mixed PSO-DE algorithm that performed well in Zhou et al. (2015), will be deployed to solve the operation optimization model of HOPs.

Online Optimization Strategy

For online task changing in oil transporting, NN models are able to predict operating parameters once the changes are given, while traditional methods should solve and iterate equations repeatedly for new conditions and inputs. To achieve online optimization, strategies based on NN identifications are as follows.

First, the optimization algorithm is connected to the Supervisory Control and Data Acquisition (SCADA) system to acquire pipeline historical and present data. These data are used to build and train NNs that can supply a more practical prediction model and faster process in solving the temperature and pressure variables. A well-trained network is seen as a specific function model that is the input of the PSO-DE algorithm. Second, keep monitoring the working

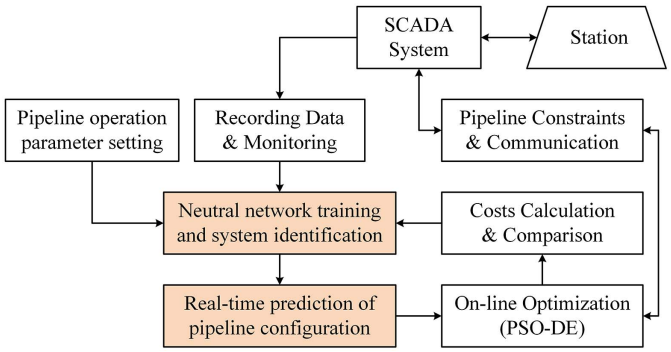

Fig. 1. Schematic of online optimization.

state of the HOPs, and when an error of the prediction model (or costs between practical and prediction) reaches a certain threshold, update the model using new recorded data and calculate a new optimization plan. Third, when the transport plan is artificially altered, the network can detect changes by itself and send new optimization results to each station through the SCADA system to execute.

The preceding strategy is based on neural network identification and prediction, which means the iteration does not need to solve complex equations about certain parameters. Once the network is trained by the recorded data, it can calculate the output easily and immediately. In addition, the SCADA system is able to collect data continuously at high sample rates and maintain real-time communication between the optimization procedures and the SCADA system, thereby guaranteeing the accuracy of the parameter prediction and the ability to react fast. A schematic of the online optimization is shown in Fig. 1.

Algorithm Descriptions of Three NNs

Traditional methods of system identification include step response identification method, frequency response method, correlation analysis method, spectral analysis method, least-squares method, and maximum likelihood method (Ljung 1999). However, though they are considerably developed, the aforementioned methods are not perfect enough for practical systems, which are usually complex, nonlinear, and uncertain. For instance, the least-squares method requires input signals that are known and varied while the exact input cannot be exactly obtained, or changing it is not allowed. When applied to nonlinear systems, such as operating HOP systems, the performance of traditional methods is not as good as that of linear ones. Traditional methods, additionally, require specific structures and parameters before being applied, which is not always practical or possible.

As smart control theories, such as neural network (Zhou et al. 2006), genetic algorithm (Halim et al. 2012), and fuzzy theory (Zhang 2003), have been studied and applied to the field of industrial control, some modern methods have been developed to solve nonlinear system identification. Methods based on neural networks are able to describe arbitrary nonlinear model, making online optimization possible for operating HOP systems. Practical data of HOP are used as input for the network, so complex modeling can be avoided and the identified result will be very close to actual results. In addition, real-time iteration can be omitted by network optimization once the training is accomplished. In this section, BPNN, RBFNN, and GRNN are applied to operating HOP system identification and their advantages are compared.

Preprocessing of Input and Output Variables

All practical data arising from pipeline historical operation are used for training a neural network. However, they cannot match their respective weights without proper scale transformation. Through scale transformation, the input and output range will be transformed and limited within $[-1, 1]$, which is also called the normalization. Scale transformation is used in cases where independent input variables have different physical meanings/units, dimensions, or orders of magnitude. When calculating the sigmoid function output of the neural network, the transformation can avoid saturation, so the matrix can be adjusted within a flat area of error surface (Stolcke et al. 2008). Otherwise, even small proportions of the total error would result in large prediction deviations. The scale transformation based on maximum $x_{\max}$ and minimum $x_{\min}$ can be changed to $[-1, 1]$ by Eqs. (7) and (8):

$$x_{\text{mid}} = \frac{x_{\max} + x_{\min}}{2} \tag{7}$$

$$\overline{x_i} = \frac{x_i - x_{\text{mid}}}{\frac{1}{2}(x_{\max} - x_{\min})} \tag{8}$$

wherein x_{mid} = median value of all samples and is transformed into 0, with a maximum value of 1 and minimum value of -1.

As noted, the input variables are listed in Eq. (2). For convenience of algorithm description, now the foregoing transformations are applied and rewritten as a standard input vector $\boldsymbol{X} = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}$, all mentioned previously as well. Likewise, the output variables mentioned in Eq. (6) will be recorded as $\boldsymbol{O} = (o_1, o_2, \ldots, o_l)^{\mathrm{T}}$.

BPNN and Its Performance

A backpropagation neural network (BPNN) is a feedforward neural network of a multilayer perceptron based on an error backward propagation (BP) algorithm. The whole training process consists of two parts: backpropagation of error and forward propagation of practical signals. When errors go back from output layer toward the input layer, each unit in the hidden layers will share the errors to adjust their weights. When sample signals go forward and the output is calculated through hidden layers, if the output error cannot meet the requirement of some preset value, it will be sent backward repeatedly until the requirement is met (Haykin 2009).

The whole network consists of three layers, with a corresponding vector in each layer. The input vector $\boldsymbol{X} = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}$ is originally assigned by historical data; the hidden layer output vector $\boldsymbol{Y} = (y_1, y_2, \ldots, y_m)^{\mathrm{T}}$ is determined by Eq. (9). The network output vector $\boldsymbol{O} = (o_1, o_2, \ldots, o_l)^{\mathrm{T}}$ is determined by Eq. (10), in which the bipolar sigmoid function $f(x)$ is defined by Eq. (11). The expected network output $\boldsymbol{d} = (d_1, d_2, \ldots, d_l)^{\mathrm{T}}$, which is determined by the pipeline history data, is applied to control the required error. The matrices $\boldsymbol{W} = (\boldsymbol{W}_1, \boldsymbol{W}_2, \ldots, \boldsymbol{W}_l)$ and $\boldsymbol{V} = (\boldsymbol{V}_1, \boldsymbol{V}_2, \ldots, \boldsymbol{V}_m)$ transfer weights that are initialized randomly between layers:

$$y_i = f\left(\sum_{j=0}^{m} \nu_{ij} x_i\right), \quad j = 1, 2, \ldots, m \tag{9}$$

$$o_k = f\left(\sum_{j=0}^{m} \omega_{jk} y_i\right), \quad k = 1, 2, \ldots, l \tag{10}$$

$$f(x) = \frac{1 - e^{-x}}{1 + e^{-x}} \tag{11}$$

The standard BPNN algorithm steps for the optimization of HOP operations are given in what follows:

Step 1: Initialize network. Assign $\boldsymbol{W}$, $\boldsymbol{V}$ with random values, generate training counter q and pattern counter p to 1. Then set the error $E = 0$, network precision $E_{\min}$, and learning efficiency η according to practical requirements.

Step 2: Input vector $\boldsymbol{X}$ and $\boldsymbol{d}$ with pipeline history data, calculate $\boldsymbol{Y}$, $\boldsymbol{O}$ using Eqs. (9) and (10).

Step 3: Calculate network error. Given that the quantity of the training sample is p, the error of the sample network is calculated by Eq. (12), and the overall error (E_{RME}) is calculated by Eq. (13):

$$E^p = \sqrt{\sum_{k=1}^{l} (d_k^p - o_k^p)^2} \tag{12}$$

$$E_{RME} = \sqrt{\frac{1}{p}\sum_{p=1}^{P} (E^p)^2} \tag{13}$$

Step 4: Calculate errors of every layer, δ_k^o and δ_j^y, using Eqs. (14) and (15):

$$\delta_k^o = (d_k - o_k)o_k(1 - o_k) \tag{14}$$

$$\delta_j^y = \left(\sum_{k=1}^{l} \delta_k^o \omega_{jk}\right) y_i(1 - y_i) \tag{15}$$

Step 5: Adjust weights between layers using Eqs. (16) and (17):

$$\Delta\omega_{jk} = \eta\delta_k^o y_i = \eta(d_k - o_k)o_k(1 - o_k)y_i \tag{16}$$

$$\Delta\nu_{ij} = \eta\delta_j^y x_i = \eta\left(\sum_{k=1}^{l} \delta_k^o \omega_{jk}\right) y_i(1 - y_i)x_i \tag{17}$$

Step 6: Determine whether all samples are deployed in training. If condition $p < P$ is satisfied, go to Step 2; otherwise go to Step 7.

Step 7: If the total error E_{RME} satisfies the condition $E_{RME} < E_{\min}$, then the training process ends, else generate $E = 0$ and $p = 1$, then go back to Step 2.

Figs. 2 and 3 indicate how the number of samples and the node number of the hidden layer influence the identification results. The sample data are collected by the SCADA system and transformed to standard input vector. It can be seen that as the sample number

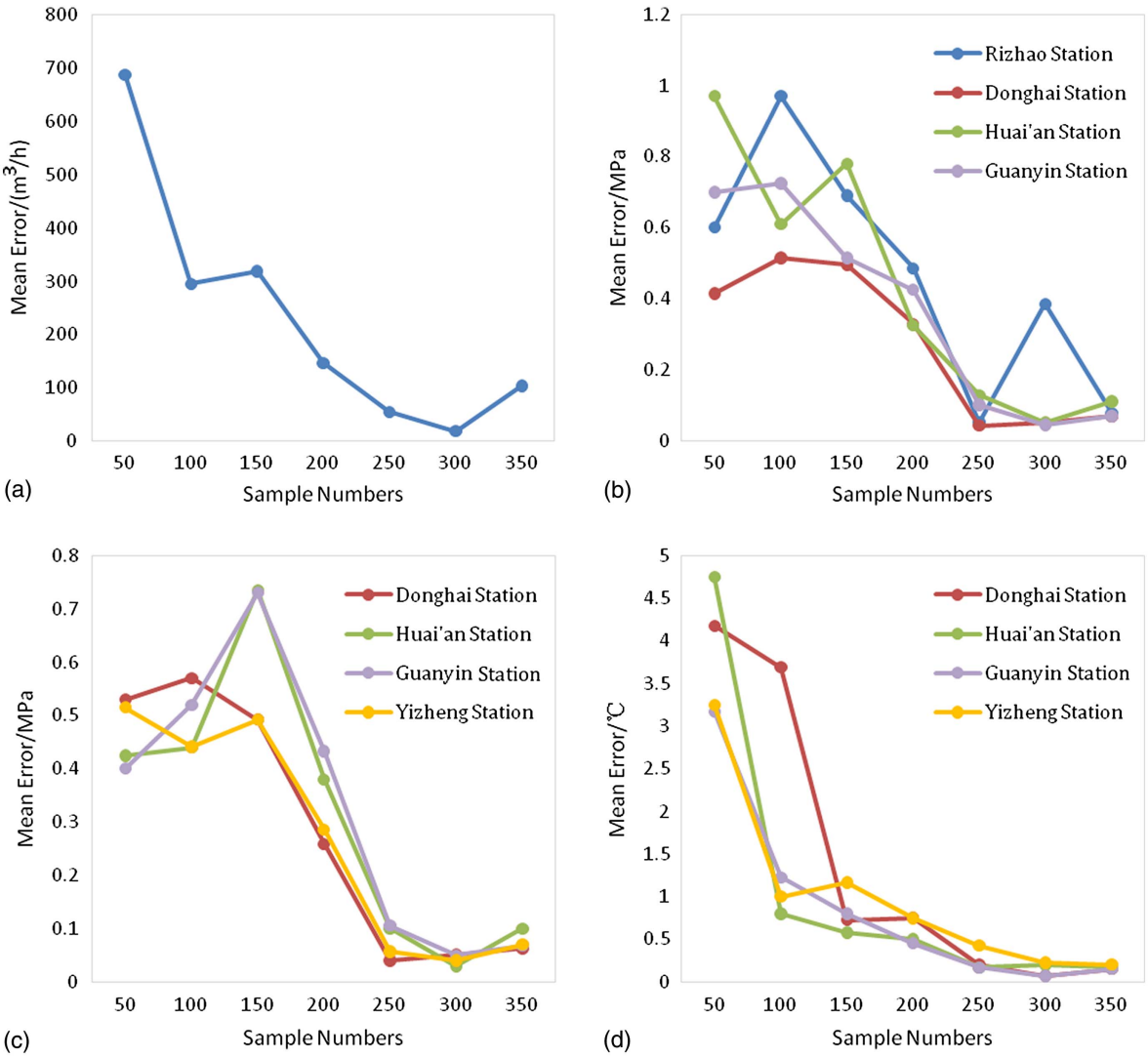

Fig. 2. Influences of the sample number on the BP NN identification performances: (a) flow; (b) output pressure; (c) input pressure; and (d) input temperature.

J. Pipeline Syst. Eng. Pract., 2020, 11(1): 04019040

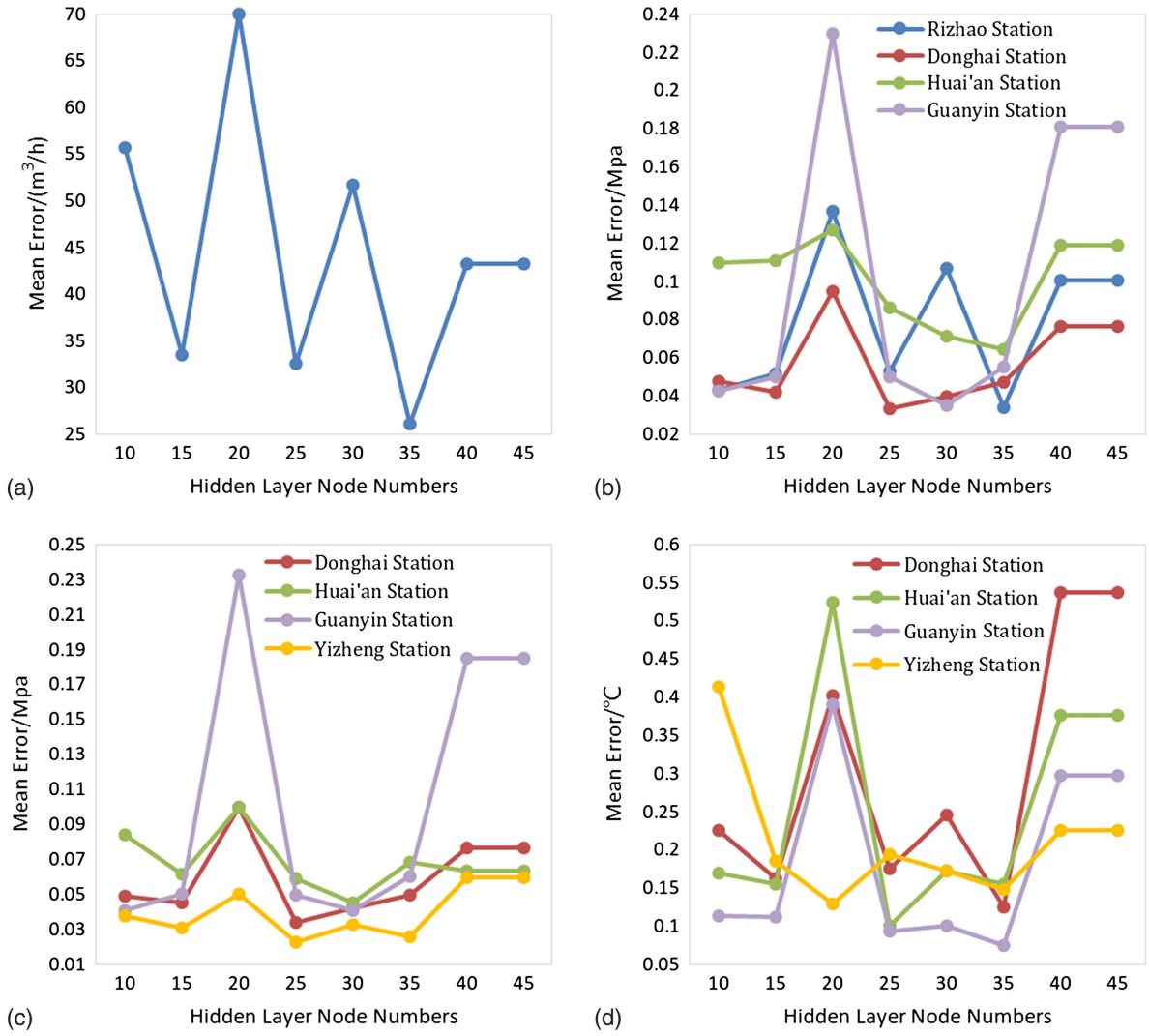

Fig. 3. Influences of the node number of the hidden layer on BPNN identification performances: (a) flow; (b) output pressure; (c) input pressure; and (d) input temperature.

increases, identification errors of all 13 working parameters decrease. When the number reaches 250, the identification error is small enough to be acceptable, and further increasing the sample number does not demonstrate significant reduction of errors. However, despite the fact that the errors of the flow and outlet pressure at the Rizhao station is evidently abnormal, the errors of the other 11 parameters go up first and then go down to a minimum, while the hidden layer's node number is between 25 and 35, and after that the error increases again.

RBFNN and Its Performance

A radial basis function neural network (RBFNN) is usually used as a local approximation network. It is unlike the BPNN, whose global output can be influenced by any single change of the weights in the whole network. The RBFNN is influenced by local weights and therefore allows for a quicker study and training process. For a RBFNN model, $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}$ is the input vector, $\boldsymbol{y} = (y_1, y_2, \ldots, y_m)^{\mathrm{T}}$ is the output vector, $\boldsymbol{W} \in \mathbb{R}^{h \times m}$ is the matrix of the output weights, $\phi_i(*)$ is the activation function of hidden layer units where i describes the index of the independent units, and the function $\phi_i(*)$ can be defined by three kinds of radial basis function as Eqs. (18)–(20). The radial basis function has better local adjustability that when the nerve cell is further from the central point in an n-dimensional space, the neuron will be activated to a less extent according to Feng et al. (2009).

Gaussian function:

$$\phi_i(t) = e^{-\frac{t^2}{\delta_i^2}} \tag{18}$$

Reflected sigmoidal function:

$$\phi_i(t) = \frac{1}{1 + e^{-\frac{t^2}{\delta_i^2}}} \tag{19}$$

Inverse multiquadric function:

$$\phi_i(t) = \frac{1}{(t^2 + \delta_i^2)^{\alpha}}, \quad \alpha > 0 \tag{20}$$

Experiments show that the inverse multiquadric function method produces the least error, so the inverse multiquadric function is chosen as the activation function. The supervised learning method (Tan et al. 1995) is used to train parameters in RBFNNs. The objective function is defined as Eq. (21):

$$E = \frac{1}{2}\sum_{i=1}^{P} e_i^2 \tag{21}$$

where P is the total number of training samples; and e_i is the error of the ith input sample, which is calculated by Eq. (22):

$$e_i = d_i - F(x_i) = d_i - \sum_{j=1}^{M} \omega_j G(\|x_i - c_i\|) \tag{22}$$

Additionally, in the applied gradient descent method, all corrections of the parameters should be proportional to their negative gradients to minimize the value of the objective function, as shown in Eqs. (23)–(25):

$$\Delta c_j = -\eta \frac{\partial E}{\partial c_j} = \eta \frac{\omega_j}{\delta_j^2} \sum_{i=1}^{P} e_i G(\|x_i - c_i\|)(x_i - c_i) \tag{23}$$

$$\Delta \delta_j = -\eta \frac{\partial E}{\partial \delta_j} = \eta \frac{\omega_j}{\delta_j^2} \sum_{i=1}^{P} e_i G(\|x_i - c_i\|)\|x_i - c_i\|^2 \tag{24}$$

$$\Delta \omega_j = -\eta \frac{\partial E}{\partial \omega_j} = \eta \frac{\omega_j}{\delta_j^2} \sum_{i=1}^{P} e_i G(\|x_i - c_i\|) \tag{25}$$

The RBFNN algorithm steps for the optimization of the HOP operation are given in what follows:

Step 1: Initialize the network with random values, set the error $E = 0$, and set the network precision $E_{\min}$ and the learning efficiency η according to practical requirements.

Step 2: Construct vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ using the pipeline historical data, and calculate the error of the overall network using Eqs. (21) and (22).

Step 3: For each single output of the RBFNN, calculate the gradients and adjust the errors using Eqs. (23)–(25).

Step 4: If the total error E_{RME} satisfies the condition $E_{RME} < E_{\min}$, then the training process ends, else generate $E = 0$, then go back to Step 2 and adjust the local single RBFNN.

Fig. 4 Illustrates that, as the sample number increases, all identification errors have a decreasing tendency using the RBFNN method. When the sample number reaches 250 or more, similarly, the identification error is small enough to be acceptable, and further increasing the sample number does not demonstrate significant results.

GRNN and Its Performance

The general regression neural network (GRNN) was proposed by Specht in 1991 (Specht 1991). GRNN deploys a continuous joint

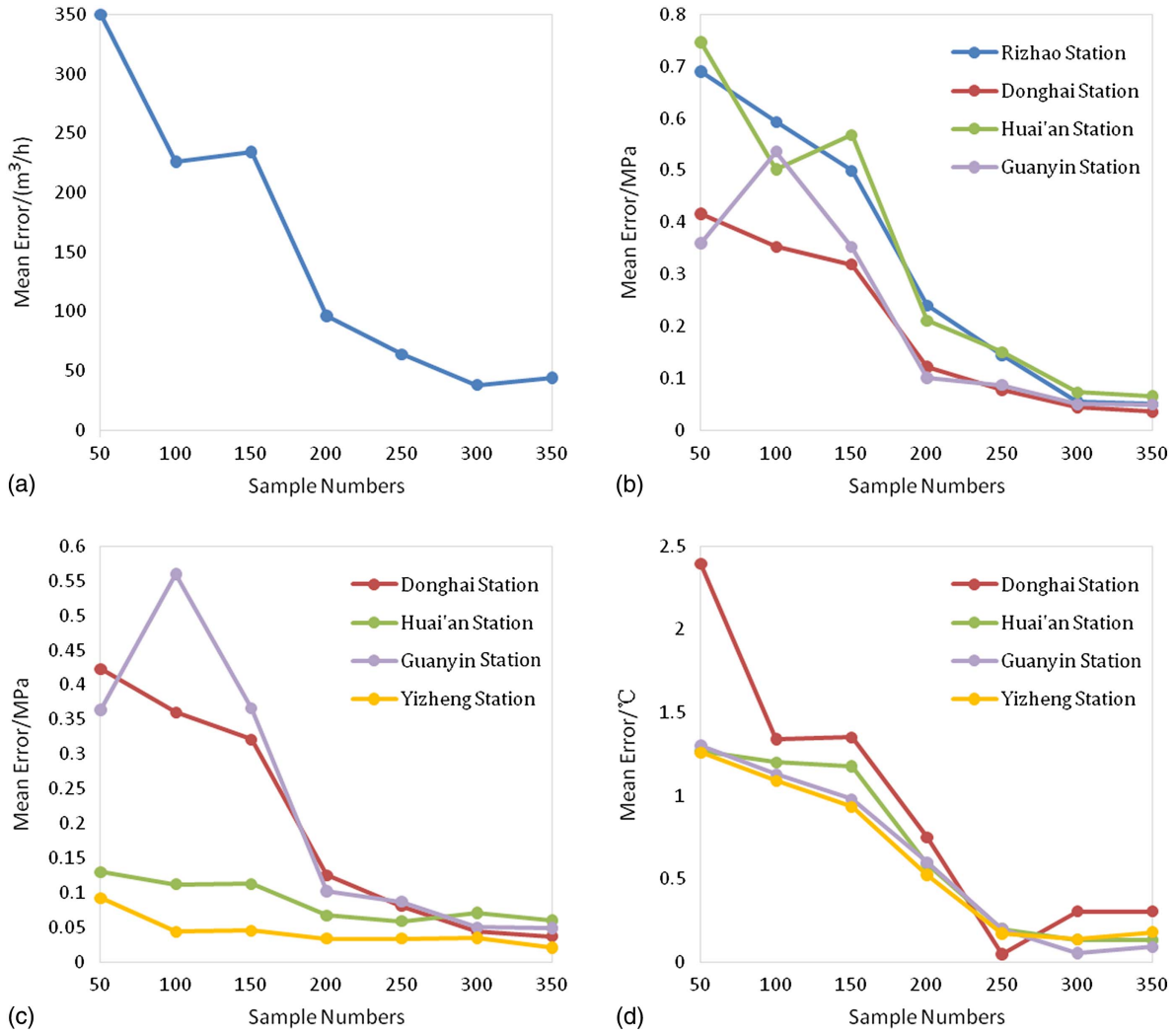

Fig. 4. Influences of the sample number on RBFNN identification performances: (a) flow; (b) output pressure; (c) input pressure; and (d) input temperature.

probability density function (JPDF) instead of a fixed function. The GRNN acquires a JPDF of dependent and independent variables by observation samples and then executes a Parzen nonparametric estimation (Singh and Karthikeyan 2012), thereby calculating the regression value of the dependent variables to the independent variables as a consequence, wherein there is a smooth factor σ in the kernel function of the regression unit, which has a significant impact on the estimation result and so requires optimization.

Assume that $f(\boldsymbol{x},\boldsymbol{y})$ represents the known continuous joint probability density function of a vector random variable $\boldsymbol{x}$ and a scalar random variable $\boldsymbol{y}$. Apply $\boldsymbol{x}_0$ as a particular value of $\boldsymbol{x}$; the regression of $\boldsymbol{y}$ on $\boldsymbol{x}_0$ can be calculated using Eq. (26):

$$E(y/x_0) = \hat{y}(x_0) = \frac{\int_{-\infty}^{\infty} y f(x_0, y)dy}{\int_{-\infty}^{\infty} f(x_0, y)dy} \tag{26}$$

Apply Parzen nonparametric estimation to the sample $\{x_i, y_i\}_{i=1}^{n}$, so the probability density function $f(\boldsymbol{x}_0,\boldsymbol{y})$ can be estimated by Eq. (27), where p is dimensional of $\boldsymbol{x}$, σ is the smooth factor, and n is the sample size:

$$f(x_0, y) = \frac{1}{n(2\pi)^{\frac{p+1}{2}}\sigma^{p+1}} \sum_{i=1}^{n} e^{-d(x_0,x_i)} e^{-d(y,y_i)} \tag{27}$$

$$d(x_0, x_i) = \sum_{j=1}^{P} \left(\frac{x_{0j} - x_{ij}}{\sigma}\right)^2, \quad d(y, y_i) = (y - y_i)^2 \tag{28}$$

$$\hat{y}(x_0) = \frac{\sum_{j=1}^{n} y_j e^{-d(x_0,x_i)}}{\sum_{j=1}^{n} e^{-d(x_0,x_i)}} \tag{29}$$

Synthesizing Eqs. (26) and (27) and $\int_{-\infty}^{\infty} z e^{-z^2} dz = 0$, Eq. (29) can be obtained, which describes the weighted sums of $e^{-d(x_0,x_i)}$. It can be seen that when the smooth factor σ is large enough, $d(x_0, x_i)$ approaches zero, which indicates that $\hat{y}(x_0)$ is approaching the mean of all the dependent output variables. Conversely, when σ approaches zero, $\hat{y}(x_0)$ is very close to the practical values of the samples. In the case of overfitting when $\hat{y}(x_0)$ is not included in the sample sets, σ is weighted by the distance between the samples and the estimated $\hat{y}(x_0)$.

GRNN resembles RBFNN in terms of composition structure. The input layer $\boldsymbol{x}$ consists of linear units. The second layer, however, whose nodes correspond to the sample and are activated by the Gauss function, is called the pattern layer. Moreover, the third layer is used to calculate the weighted sum of y_i. For the last layer, $\hat{y}(x_0)$ is estimated as the output of the network.

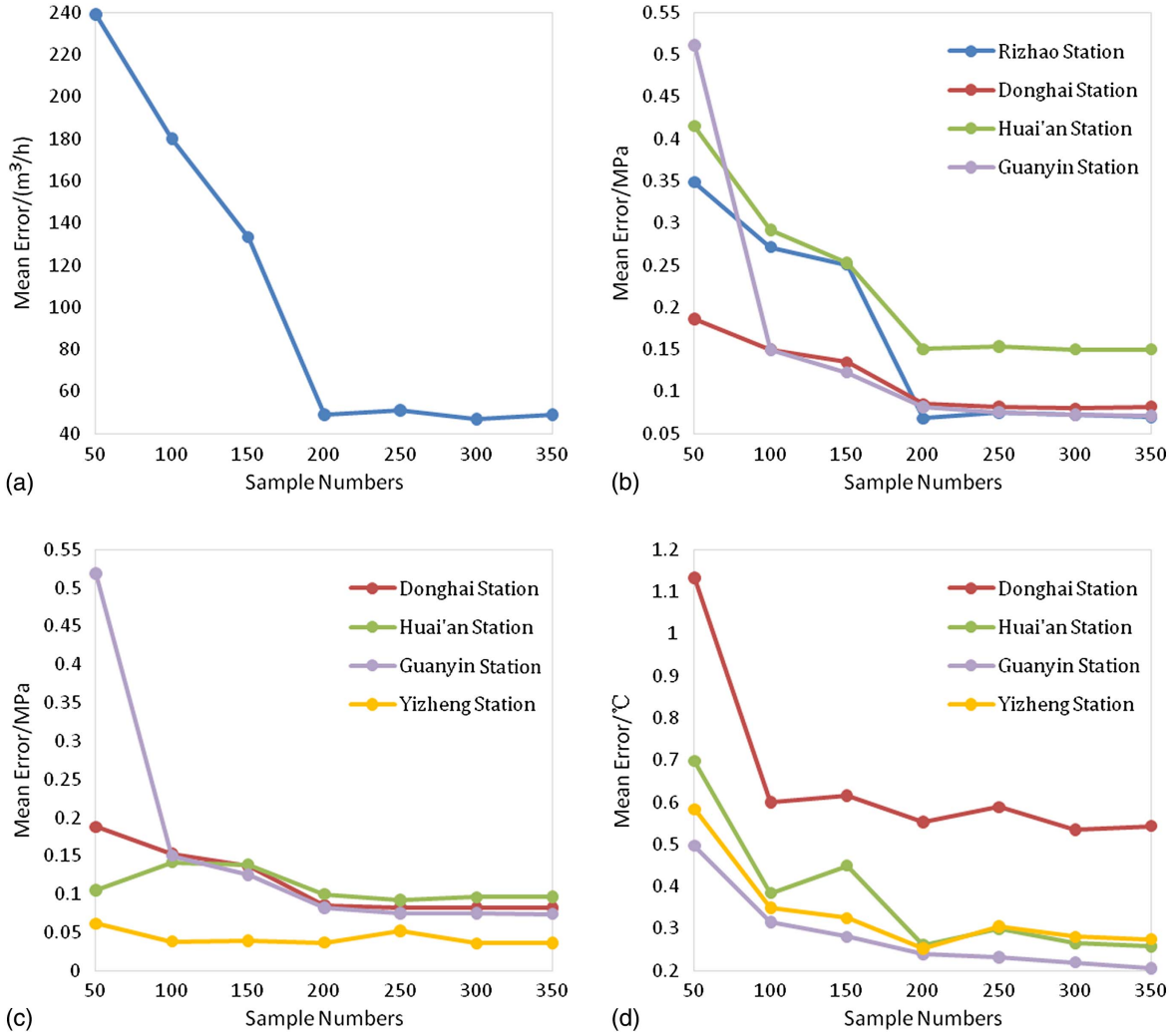

Fig. 5. Influences of the sample number on GRNN identification performances: (a) flow; (b) output pressure; (c) input pressure; and (d) input temperature.

J. Pipeline Syst. Eng. Pract., 2020, 11(1): 04019040

The ideal smooth factor σ cannot be exactly determined via the size of the sample, thus golden section search method is used to search for the best σ for a superior network. The steps are as follows (Fu et al. 2012):

1. Generate a random value for the smooth factor σ.
2. Select a sample optionally as a network error testing input, with the other samples used for training.
3. Test the error of the network using the testing input in Step 2.
4. Repeat Steps 2 and 3 until all samples are tested, calculate the average error as the objective function using Eq. (30):

$$e_{total}(\sigma) = \frac{1}{n}\sum_{i=1}^{n} |\hat{y}(x_i) - y(x_i)| \tag{30}$$

Fig. 5 demonstrates that, in comparison to BPNN and RBFNN, GRNN is much less dependent on the number of samples. As the number increases, the identification errors of all 13 working parameters generally decrease. When the sample number approaches 200 or more, the error can be considered acceptable. For inlet pressure and temperature prediction, when the sample number exceeds 100, the prediction performance is stable enough.

Table 1. Pipeline characteristics of Rihzhao-Yizheng pipelines

Pipeline section	Length (km)	Diameter (mm)	Pipe thickness (mm)	Material (steel)	Design pressure (MPa)	Volume (m^3)
Rizhao-Donghai	84.7	914	12.7	L450	8.5	52,501
Donghai-Huai'an	91.5	914	12.7	L450	8.5	56,716
Huai'an-Guanyin	94	914	12.7	L450	8.5	58,265
Guanyin-Yizheng	107.8	914	12.7	L450	8.5	66,819

Contrast Experiments of BPNN, RBFNN, and GRNN

To carry out a contrast experiment of NNs, pipeline operating data are collected by the SCADA system, which is widely used and accepted as a standard industrial data acquisition and control system. The SCADA system integrates more than pressure transducers, temperature sensor, data collection and communication devices, electrical controllers, monitoring, and a storage center. The pressure data were collected by pressure transducers at each mentioned pumping station at a real-time sample rate of 200 Hz. The flows of oil are collected from flow meters at the stations, and temperature

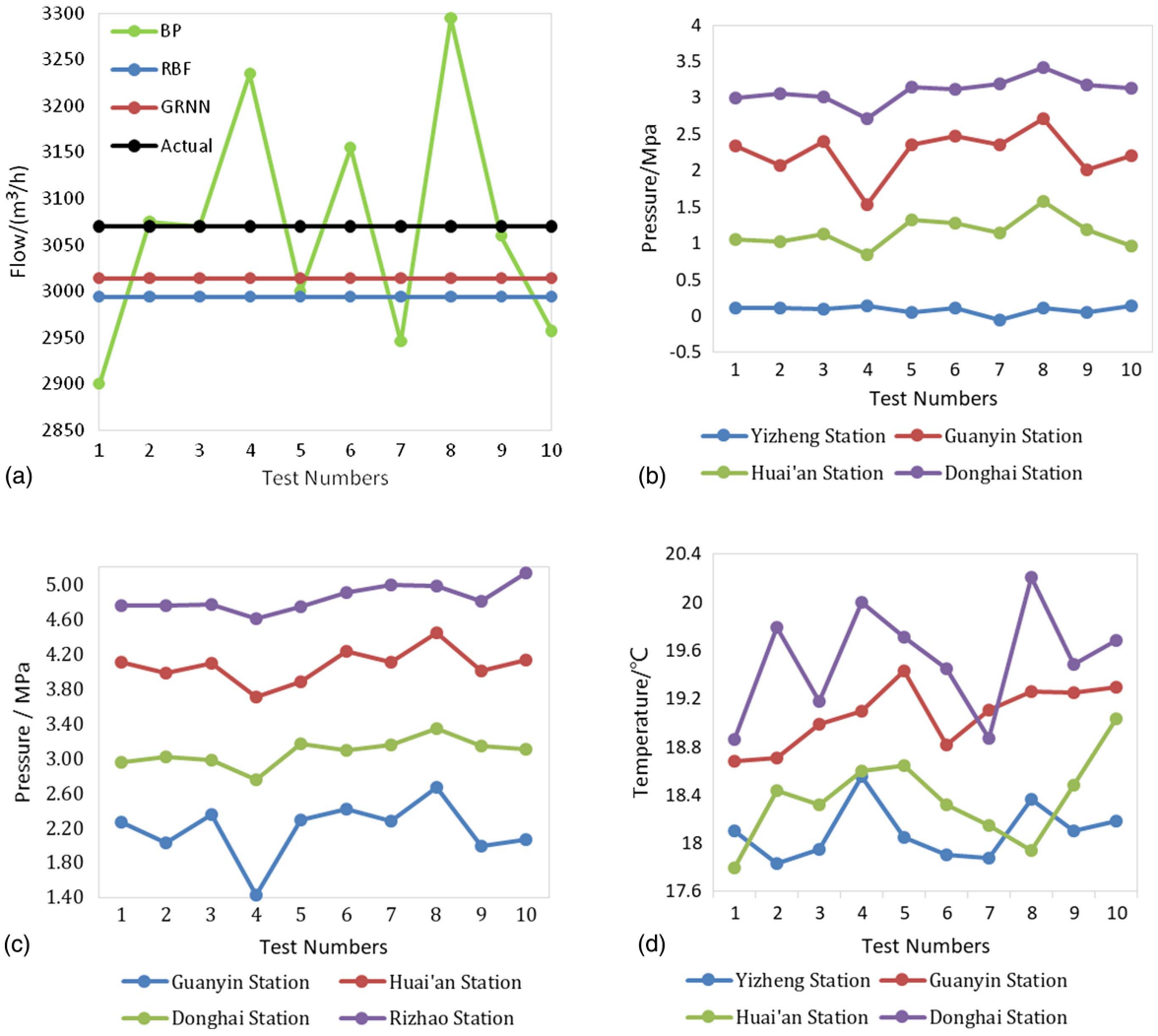

Fig. 6. (a) Contrasts of three NNs' stability for flow prediction; and 10 tests of BPNN's stability for (b) input pressure prediction; (c) output pressure prediction; and (d) input temperature prediction.

Table 2. Variance of BPNN identification in different tests

Station	Output pressure (MPa)	Input pressure (MPa)	Input temperature (°C)
Guanyin	0.099	0.092	0.060
Huai'an	0.033	0.038	0.119
Donghai	0.023	0.027	0.484
Rizhao	0.023	—	—
Yizheng	—	0.003	0.047

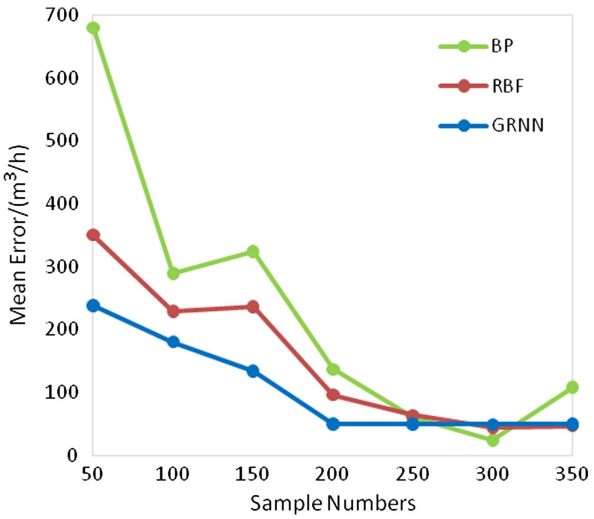

Fig. 7. Sensitivities of three NNs' identification errors to the sample number.

data were collected by thermometers at heating stations. The data were acquired and recorded in SCADA. The pipeline characteristics are listed in Table 1.

Stability Analysis

To observe the stability of BPNN, RBFNN, and GRNN, the same samples are used to carry out the network training 10 times. All 13 parameters are predicted using the 3 different networks. Taking the flow identification as an example, as shown in Fig. 6(a), for the 10 tests, the predicted values of the flow using the RBFNN and GRNN remain very steady, except that the results of using the BPNN show severe randomness and nonstationarity. The 13 variables' prediction results from 10 tests using the BPNN are shown in Figs. 6(b–d). The flow prediction errors of the RBFNN and GRNN are 60 and 70 m^3/h, accounting for 1.9% and 2.3% of the actual flow, respectively. Testing experiments demonstrate that RBFNN and GRNN yield stable results when it comes to predicting all 13 parameters. Compared with the BPNN, the RBFNN and GRNN can also provide high performance for other parameter predictions. The prediction errors of other parameters using the BPNN are presented in Table 2. The large errors indicate unstable prediction.

Fig. 7 illustrates the mean absolute errors of the variable flow show decreasing trends when the sample numbers increase, wherein RBFNNs show more stable trends and slower decreasing speed than BPNN, while the GRNN shows the most stable trend and the slowest speed among the three NNs. To characterize the sensitivity of the sample numbers, the maximum forward differences are calculated as a description of stability and the performances of the three NNs are compared in Table 3. It can be seen that for all the parameters, the BPNN has the largest forward differences, and the GRNN is obviously preferred over the other two for its error stability control.

Table 3. Maximum forward differences of 3 NNs' identification errors

Station	Output pressure (MPa)			Input pressure (MPa)			Input temperature (°C)		
	BP	RBF	GRNN	BP	RBF	GRNN	BP	RBF	GRNN
Guanyin	0.34	0.25	0.34	0.31	0.26	0.31	1.95	0.39	0.20
Huai'an	0.43	0.37	0.11	0.35	0.03	0.02	3.98	0.55	0.30
Donghai	0.28	0.19	0.05	0.24	0.19	0.05	2.95	1.00	0.51
Rizhao	0.44	0.26	0.19	—	—	—	—	—	—
Yizheng	—	—	—	0.22	0.05	0.02	2.25	0.45	0.25

Online Optimization Results and Discussions

Ten considerably different sample groups were selected for all 13 working parameters. Their outputs are predicted using pipeline historical data, and they are applied to the PSO-DE optimization algorithm. Some of the results are presented in Tables 4 and 5, wherein the input variables in Table 4, partly changeable and controllable, are to be optimized. Table 5 presents outlet variables that require real-time identification and monitoring to make sure they satisfy the constraints.

Table 4 illustrates that the scheduling scheme of pumps does not change significantly, except that the Rizhao station in Group 5 decreases from three large pumps to two large ones. While the outlet temperature at each station changes in different ways, all changes serve to fit the overall process. In Table 5 it can be seen that before optimization the highest pressure is 5.01 MPa, while after optimization the lowest is 4.6 MPa; the highest input pressure decreases from 3.45 to 3.2 MPa, and the lowest input temperature falls from 13.8°C to 12.6°C. The largest flow decrease is 174.3 m^3. Table 5 shows that all parameters, including input pressure, output pressure, input temperature, and flow, change reasonably within the constraints, so that after optimization, all costs decline, with the largest cost savings being has high as 10.75%, demonstrating that the online identification and optimization method works well on the heated oil pipeline.

The training computation time from using MATLAB (64-bit version) is less than 3 h for each of the three NNs. Using the well-trained neural network, prediction of the operation parameters of the entire pipeline can be completed in an instant, less than 1 ms. On this basis, the time consumption for finding the optimal operation parameter set using PSO will not exceed 2 min. When a new transport task is assigned, it usually takes at least a few hours for the pipeline to operate under the newly adjusted operating conditions. Optimization happens in real time.

Conclusions

1. Three different neural networks are compared to identify their working parameters. RBFNN and GRNN have smaller and steadier identification errors than BPNN, and the calculations are fast once the neural network training is completed using historical data.
2. Instead of constructing and solving the optimization model, well-trained neural networks are used in executing the PSO-DE

Table 4. Optimization results for the 13 input variables

Group	Result	Pump group		Average ground temperature between stations (°C)				Oil density (kg/m^3)	Temperature at 1st station (°C)	Outlet temperature (°C)			
		Rizhao	Huai'an	Ri-Dong	Dong-Huai	Huai-Guan	Guan-Yi			Rizhao	Donghai	Huai'an	Guanyin
1	Before	1L	1S	15.11	14.13	14.77	15.28	868.6	23.7	24.4	20.1	18.1	17.2
	After	1L	1S	15.11	14.13	14.77	15.28	868.6	23.7	22.9	19.4	18.9	16.6
2	Before	1L	1S	11.27	10.16	11.13	11.74	868.7	25.1	25.9	18.5	16.6	16.3
	After	1L	1S	11.27	10.16	11.13	11.74	868.7	25.1	25.5	16.1	18	17.5
3	Before	1L1S	1S	10.9	8.5	9.4	11.1	879.7	19.9	21.9	16.1	14.5	14.7
	After	1L1S	1S	10.9	8.5	9.4	11.1	879.7	19.9	22.1	16.7	13.1	14.4
4	Before	1L1S	1S	10.9	8.4	9.5	10.9	877	21.2	22.1	14.8	14.1	14.8
	After	1L1S	1S	10.9	8.4	9.5	10.9	877	21.2	22.6	14.2	15.5	13.9
5	Before	3L	1L	18.2	15.2	16.3	16	859.8	26	26.7	24	22.6	21.9
	After	2L	1L	18.2	15.2	16.3	16	859.8	26	28.1	23.8	21.7	19.3
6	Before	2L	1L1S	11.71	10.56	11.48	12.04	858.5	22.9	24	20.6	18.2	16.8
	After	2L	1L1S	11.71	10.56	11.48	12.04	858.5	22.9	23.8	19.2	17.1	15.9
7	Before	1L	1L1S	11.61	10.46	11.43	11.98	868.7	23.1	24.2	19.5	17.5	16.6
	After	1L	1L1S	11.61	10.46	11.43	11.98	868.7	23.1	23.6	19.4	18.1	14.3
8	Before	2L	1L1S	11.2	7.5	9.1	11.4	877.4	21.6	22.8	17.5	17.1	17.3
	After	2L	1L1S	11.2	7.5	9.1	11.4	877.4	21.6	22.2	16.8	17.7	16.2
9	Before	2L	1L1S	11.8	9.6	10.1	11.5	879.2	22.3	23.3	18.5	16.7	15.5
	After	2L	1L1S	11.8	9.6	10.1	11.5	879.2	22.3	24.1	18.3	17.3	15.3
10	Before	2L	1L1S	12.78	11.46	12.2	12.82	856.4	23.6	24.7	20.3	18.6	17.6
	After	2L	1L1S	12.78	11.46	12.2	12.82	856.4	23.6	24.3	19.2	18.1	16.6

Note: L = large; and S = small.

Table 5. Optimization results of the 13 output variables

Group	Result	Output pressure (MPa)				Input pressure (MPa)				Input temperature (°C)				Flow (m^3/h)	Fees (¥/h)
		Rizhao	Donghai	Huai'an	Guanyin	Donghai	Huai'an	Guanyin	Yizheng	Donghai	Huai'an	Guanyin	Yizheng		
1	Before	3.12	2.26	2.36	1.36	2.28	1.2	1.39	0.06	20.2	17.7	17.2	15.9	2,231	3,576
	After	2.8	2.4	2.3	1.2	2.2	1.1	1.4	0.1	19.4	17.6	17.5	16.3	2,183.6	3,244
2	Before	3.23	2.32	2.45	1.41	2.33	1.23	1.44	0.09	18.6	16.1	16.4	15.3	2,550	3,909
	After	3.12	2.3	2.15	1.5	2.21	0.9	1.5	0.06	17.7	15.2	15.9	14.7	2,450.6	3,513
3	Before	4.2	2.98	2.72	1.53	3.01	1.6	1.56	0.04	16.3	13.8	14.8	13.9	2,338	4,725
	After	3.1	2.2	2.8	2.6	2.9	1.2	1.5	0.1	15.1	12.2	13.5	12.6	2,260.5	4,283
4	Before	4.29	3.07	2.92	1.64	3.1	1.82	1.67	0.06	15	13.6	14.7	13.8	2,439	5,032
	After	4.15	3.12	2.2	1.56	3	1.5	1.4	0.1	13.9	12.3	13	13.1	2,361.7	4,491
5	Before	4.75	3.2	3.45	1.93	3.23	1.44	1.97	0.1	24.2	21.3	22	20.2	2,751	7,216
	After	4.3	3.1	3.1	1.7	2.9	1.1	1.9	0.04	24.1	21	20.6	18.6	2,621.5	6,809
6	Before	4.76	3.41	4.43	2.41	3.45	1.39	2.47	0.07	20.7	17.4	16.8	15.6	2,879	7,112
	After	4.1	2.8	3.78	2.1	2.9	1.1	2.2	0.13	19.6	16.7	17.2	16.3	2,831.8	6,782
7	Before	4.86	3.17	4.26	2.3	3.21	1.21	2.36	0.1	19.6	16.7	16.6	15.2	2,865	7,014
	After	4.2	2.3	3.3	1.3	2.4	0.9	1.3	0.1	18.9	15.7	16.1	14.9	2,690.7	6,594
8	Before	4.8	3.03	4.09	2.25	3.06	1.29	2.29	0.09	17.7	16.2	17.3	15.8	3,036	7,311
	After	4.2	2.9	3.7	2.1	2.9	1.1	2.2	0.07	17.1	15.3	16.4	15.4	2,883.4	6,843
9	Before	5.01	3.34	4.54	2.51	3.37	1.38	2.55	0.14	18.7	15.6	15.6	14.5	2,819	7,244
	After	4.6	3.0	3.8	2.1	3.1	1.0	2.2	0.07	18.5	15.3	14.4	13.7	2,712.1	6,792
10	Before	4.69	3.12	4	2.16	3.16	1.21	2.22	0.11	20.3	17.8	17.5	16.1	3,114	7,757
	After	4.4	3	3.8	1.9	3.2	1.2	2.1	0.07	19.7	17.1	16.2	15.6	2,948.1	7,143

optimization algorithm. The optimization results are applied to the Rizhao–Yizheng digital long oil pipeline. Experiments shows that, if properly assigned, most of the prediction errors of the flow, outlet pressure, and inlet temperature can be controlled below 50 m^3/h, 0.1 MPa, and 0.1°C, respectively, so online optimization is achievable. The maximum reduction of energy costs in the experiment is 10.75%, meaning the savings are significant.

Acknowledgments

This work is supported by National Natural Science Foundation of China (51604192, 61773283, and 61473205) and the China Postdoctoral Science Foundation (2018M630271).

References

Abdi, H., S. D. Beigvand, and M. L. Scala. 2017. "A review of optimal power flow studies applied to smart grids and microgrids." *Renewable Sustainable Energy Rev.* 71 (May): 742–766. https://doi.org/10.1016/j.rser.2016.12.102.

Behera, S., S. Sahoo, and B. B. Pati. 2015. "A review on optimization algorithms and application to wind energy integration to grid." *Renewable Sustainable Energy Rev.* 48 (Aug): 214–227. https://doi.org/10.1016/j.rser.2015.03.066.

Botros, K. K., D. Sennhauser, K. J. Jungowski, G. Poissant, H. Golshan, and J. Stoffregen. 2004. "Multi-objective optimization of large pipeline networks using genetic algorithm." In Vol. 3 of *Proc., Biennial Int. Pipeline Conf.*, 2005–2015. New York: ASME.

Feng, Y., Z. F. Wu, J. Zhong, C. X. Ye, and K. G. Wu. 2009. "An enhanced swarm intelligence clustering-based RBF neural network web text classifier, advances in neural networks." In *Proc., 6th Int. Symp. on Neural Networks*, 684–693. New York: Springer.

Fu, X., X. Xue, and H. Li. 2012. "Application of GRNN model in predicting coal and gas outburst and gas content." *Zhongguo Anquan Kexue Xuebao* 22 (1): 24–28.

Gao, S., Y. Wang, and P. Xu. 2004. "Application of hybrid genetic algorithm in optimal operation of oil pipeline." *Oil Gas Storage Transp.* 23 (7): 34–37.

Gopal, V. N. 1980. "Optimizing pipeline operations." *J. Pet. Technol.* 32 (11): 2063–2067. https://doi.org/10.2118/8416-PA.

Halim, S. A., A. Ahmad, N. M. Noh, and A. M. Ali. 2012. "A development of snake bite identification system (N'viteR) using Neuro-GA." In Vol. 1 of *Proc., 2012 Int. Symp. on Information Technology in Medicine and Education (ITME 2012)*, 490–494. New York: IEEE Computer Society.

Haykin, S. 2009. *Neural networks and learning machines*. 3rd ed., 81–89. Beijing: China Machine Press.

Ilich, N., and S. P. Simonovic. 1998. "Evolutionary algorithm for minimization of pumping cost." *J. Comput. Civ. Eng.* 12 (4): 232–240. https://doi.org/10.1061/(ASCE)0887-3801(1998)12:4(232).

Kurak, S. D. 1989. "Power optimization yields savings for products pipeline." *Oil Gas J.* 87 (26): 65–66.

Lervik, J. K., Ø. Iversen, and K. T. Solheim. 2018. "Optimizing electrical heating system of subsea oil production pipelines." In *Proc., Int. Offshore and Polar Engineering Conf.*, 108–113. Mountain View, CA: International Society of Offshore and Polar Engineers.

Li, C., J. Wang, X. Wu, and W. Jia. 2011. "Operation optimization of heated oil transportation pipeline, ICPTT 2011: Sustainable solutions for water, sewer, gas, and oil pipelines." In *Proc., Int. Conf. on Pipelines and Trenchless Technology*, 733–743. Reston, VA: ASCE.

Li, K., Y. C. Wang, and Z. Yang. 2007. "Application of PSO in optimal design of hot oil pipeline." *J. Southwest Pet. Univ.* 29 (3): 150–153.

Ljung, L. 1999. *System identification: Theory for the user*. 2nd ed. Upper Saddle River, NJ: Prentice Hall.

Makaremi, Y., A. Haghighi, and H. R. Ghafouri. 2017. "Optimization of pump scheduling program in water supply systems using a self-adaptive NSGA-II: A review of theory to real application." *Water Resour. Manage.* 31 (4): 1283–1304. https://doi.org/10.1007/s11269-017-1577-x.

Oliveira, F., P. M. Nunes, R. Blajberg, and S. Hamacher. 2016. "A framework for crude oil scheduling in an integrated terminal-refinery system under supply uncertainty." *Eur. J. Oper. Res.* 252 (2): 635–645. https://doi.org/10.1016/j.ejor.2016.01.034.

Pavlov, A., I. Plohov, S. Drozdov, and V. Smirnov. 2017. "Study and optimization of electric heating for oil and petroleum products pipeline." In Vol. 3 of *Proc., 11th Int. Scientific and Practical Conf.*, 242–248. Rēzekne, Latvia: Rezekne Academy of Technologies.

Simpson, A. R., G. C. Dandy, and L. J. Murphy. 1994. "Genetic algorithms compared to other techniques for pipe optimization." *J. Water Resour. Plann. Manage.* 120 (4): 423–443. https://doi.org/10.1061/(ASCE)0733-9496(1994)120:4(423).

Singh, P. K., and S. Karthikeyan. 2012. "Combining GRNN and SVM using receiver operating characteristics (ROC) for improved classification of non coding RNA." In *Proc., 2012 Int. Conf. on Biomedical Engineering and Biotechnology (iCBEB)*, 115–118. New York: IEEE Computer Society.

Specht, D. F. 1991. "General regression neural network." *IEEE Trans. Neural Networks* 2 (6): 568–576. https://doi.org/10.1109/72.97934.

Stolcke, A., S. Kajarekar, and L. Ferrer. 2008. "Nonparametric feature normalization for SVM-based speaker verification." In *Proc., IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, 1577–1580. New York: IEEE Computer Society.

Tan, S., J. Hao, and J. Vandewalle. 1995. "A New learning algorithm for RBF neural networks with applications to nonlinear system identification." In Vol. 3 of *Proc., IEEE Int. Symp. on Circuits and Systems*, 1708–1711. New York: IEEE Computer Society.

Twaha, S., and M. A. Ramli. 2018. "A review of optimization approaches for hybrid distributed energy generation systems: Off-grid and grid-connected systems." *Sustainable Cities Soc.* 41 (Aug): 320–331. https://doi.org/10.1016/j.scs.2018.05.027.

Viani, F., F. Robol, M. Salucci, and R. Azaro. 2017. "Automatic EMI filter design through particle swarm optimization." *IEEE Trans. Electromagn. Compat.* 59 (4): 1079–1094.

Wang, Y., S. You, H. Zhang, X. Zheng, W. Zheng, Q. Miao, and G. Lu. 2017. "Thermal transient prediction of district heating pipeline: Optimal selection of the time and spatial steps for fast and accurate calculation." *Appl. Energy* 206 (Nov): 900–910. https://doi.org/10.1016/j.apenergy.2017.08.061.

Yan, H. U. A. N. G., and W. A. N. G. Peng. 2016. "Data center energy cost optimization in smart grid: A review." *J. Zhejiang Univ. (Eng. Sci.)* 50 (12): 2386–2399.

Yong, T. R., and J. T. Jefeerson. 1961. "Shell pipeline calls it dynamic programming." *J. Oil Gas* 59 (19): 8–12.

Zhang, H., Y. Liang, Q. Liao, Y. Shen, and X. Yan. 2018. "A self-learning approach for optimal detailed scheduling of multi-product pipeline." *J. Comput. Appl. Math.* 327 (Jan): 41–63. https://doi.org/10.1016/j.cam.2017.05.040.

Zhang, H., Y. Liang, J. Ma, Y. Shen, X. Yan, and M. Yuan. 2017. "An improved PSO method for optimal design of subsea oil pipelines." *Ocean Eng.* 141 (Sep): 154–163. https://doi.org/10.1016/j.oceaneng.2017.06.023.

Zhang, Y. 2003. "Identification of dynamic system based on dynamic fuzzy neural network." *J. Cent. South Univ. Technol.* 34 (3): 277–280.

Zhao, W., T. H. Beach, and Y. Rezgui. 2016. "Optimization of potable water distribution and wastewater collection networks: A systematic review and future research directions." *IEEE Trans. Syst. Man Cybern.: Syst.* 46 (5): 659–681. https://doi.org/10.1109/TSMC.2015.2461188.

Zhou, F., J. Cao, J. Zhang, X. Yin, S. Jia, and W. Zeng. 2006. "Prediction model of rolling force for tandem cold rolling mill based on neural networks and mathematical models." *J. Cent. South Univ. (Sci. Technol.)* 37 (6): 1155–1160.

Zhou, M., Y. Zhang, and S. Jin. 2015. "Dynamic optimization of heated oil pipeline operation using PSO-DE algorithm." *Meas.: J. Int. Meas. Confederation* 59: 344–351. https://doi.org/10.1016/j.measurement.2014.09.071.

Zuo, L., C. Wu, S. Liu, Y. Jiang, and X. Zhang. 2018. "Predicting monthly energy consumption of crude oil pipelines using process simulation and optimization." In *Proc., 12th Int. Pipeline Conf.—Proc., Biennial Int. Pipeline Conf.*, 3. New York: ASME.