Research Article
An Intelligence Optimized Rolling Grey Forecasting Model Fitting to Small Economic Dataset

Li Liu,1,2 Qianru Wang,2,3 Ming Liu,4 and Lian Li5

1 School of Computing, National University of Singapore, Singapore 117417
2 School of Information Science and Engineering, Lanzhou University, Lanzhou 730000, China
3 Department No. 68028, Lanzhou Military Region, Lanzhou 730058, China
4 Faculty of Computer and Information Science, Southwest University, Chongqing 400715, China
5 Department of Computer Science and Technology, HeFei University of Technology, HeFei 230009, China

Correspondence should be addressed to Li Liu; dcsliuli@gmail.com

Received 10 October 2013; Accepted 9 April 2014; Published 28 April 2014

Academic Editor: Jaeyoung Chung

Copyright © 2014 Li Liu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Grey system theory has been widely used to forecast the economic data that are often highly nonlinear, irregular, and nonstationary. The size of these economic datasets is often very small. Many models based on grey system theory could be adapted to various economic time series data. However, some of these models did not consider the impact of recent data or the effective model parameters that can improve forecast accuracy. In this paper, we proposed the PRGM(1,1) model, a rolling mechanism based grey model optimized by the particle swarm optimization, in order to improve the forecast accuracy. The experiment shows that PRGM(1,1) gets much better forecast accuracy among other widely used grey models on three actual economic datasets.

1. Introduction

Forecasting can be an important issue to many fields of economy; especially its accuracy was ensured to do a reasonable prediction that could change the economic policy of large companies and governments and ensure a more reasonable behavior by the financial actors. The ideal state is that the prediction error tends to be more and more smaller, but in fact, we can only do our best to research and develop the prediction algorithm as much as possible to improve the prediction accuracy.

Many forecasting models have been proposed; in general, these models can be divided into two categories: causal models and time-series models [1]. Causal models assume that historical relationship between dependent and independent variables will remain valid in future. Causal models include multiple linear regression analysis and econometric models which assume that independent variables could explain the variations in dependent variable. However, the limitation of causal models is the availability and reliability of independent variables.

Time-series models assume that history will repeat itself and its prediction refers to the process by which the future values of a system are forecasted based on the information obtained from the past and current data points. In the literature, two main techniques for time series prediction are statistical and artificial intelligence (soft computing) based approaches. The well-known statistical models proposed include AR (autoregressive), MA (moving average), ARMA (autoregressive moving average), ARIMA (autoregressive integrated moving average), and Box-Jenkins models. The statistical models are too weak to solve the nonlinear problems and too complex to be used in predicting future values of a time series.

The widely used artificial intelligence approaches include neural network (NN) [2–4], support vector machines (SVM) [5–8], fuzzy systems [9], linear regression, Kalman filtering [10], and hidden Markov models (HMM) [3]. All of these approaches are used for updating the model parameters. In the recent years, several hybrid models [11–14] were proposed to improve the forecast accuracy. However, these artificial intelligence based approaches demand a great deal
of training data and relatively long training period for robust generalization [11]. For those economic predictions, which are very difficult to construct a model by using neither the conventional linear statistical methods nor the artificial neural networks because the economic time series are highly nonlinear, highly irregular, and highly nonstationary [15].

Grey system theory was introduced and developed by Deng back in 1989 to be used for mathematical analysis on the phenomenon of uncertainty and roughness. It requires a small set of training data, which are discrete or incomplete, to construct a model for future forecast. The uncertainty and roughness training data are “grey” data [16]. Similarly, “white” data means that the information is completely clear, while “black” indicates that the information is completely unclear.

Grey system theory has been widely and successfully used to forecast all kinds of data in the many areas such as economic, financial, agricultural, and industrial areas and energy. In the past few years, grey system theory has been employed for solving the forecasting economic problems. The model GM(1,1) built from grey system theory has shown that this approach is very efficient to forecast the irregular and nonlinear economic time series data. A combination of residual modification and residual artificial neural network (ANN) sign estimation is proposed to improve the accuracy of the original GM(1,1) model [17–19]. However, this approach needs long training period.

Rolling mechanism is one of the most effective methods to improve the performance of grey system model and handle noisy data [7, 20–22]. The authors in [22] used the rolling mechanism to improve the forecast accuracy of grey model for education expenditure. Zhao et al. [23] proposed rolling mechanism to forecast the per capita annual net income of rural households in China and showed that it outperformed other traditional grey prediction models and a differential evolution algorithm proposed to optimize rolling grey prediction model. The authors in [24–26] proposed an improved rolling grey model, which can update the model parameters on the coal production forecast and semiconductor industry production forecast, respectively.

However, although these improved rolling mechanism based grey models could adapt to various economic time series data because they considered the recent data that can improve forecast accuracy in future prediction, they did not consider the impact of their model parameters which are fixed through the whole prediction period or only considered a simple change of the model parameters for the prediction which could perform well on noiseless sequence, but it could not adapt to the noisy data.

In this paper, we proposed an improved rolling mechanism based grey model optimized by the particle swarm optimization (PSO for short) to improve the forecast accuracy, especially for the highly irregular and noiseless data. PSO, which belongs to swarm intelligence methods, is considered as a tool for modeling behavior and for optimization of difficult numerical solutions, since it was developed by [27] as an evolutionary computing technology. PSO algorithm had been enormously successful on about 700 applications [28]. We choose PSO to optimize our model parameters for two significant reasons: its routinely delivering of good optimization results like NN methods and its simplicity to get better results in a faster and cheaper way that NN methods cannot achieve.

This paper examines a rolling mechanism based grey model with PSO optimization on economic data. Section 2 outlines the original grey model GM(1,1) and the improved GM(1,1) model with rolling mechanism. Section 3 presents the rolling mechanism based grey model with PSO optimization. We also propose a PSO based algorithm that searches the best value for the model parameter. Furthermore, we illustrate that our model gets much better performance on three economic dataset: financial intermediation in Beijing, real estate in Beijing, and semiconductor production in Taiwan, compared with other grey system theory based models. Section 5 concludes this paper.

2. Grey Model Background

The grey system theory mainly focuses on extracting realistic governing laws of the system from the available data of the system generally with white noise data. A grey model in grey system theory is denoted by GM(n,m), where n indicates the order of the difference equation and m indicates the number of variables.

GM(1,1) is the original grey model, which has been widely applied to carry on the short-term prediction because of its computational efficiency. It uses a first order differential equation to predict an unknown system. A GM(1,1) algorithm is described below.

**Step 1.** The original time sequence is initiated by

\[ x^{(0)} = (x^{(0)}(1), x^{(0)}(2), \ldots, x^{(0)}(n)) \]

where \( x^{(0)}(i) \) the time series data at time \( i \) and \( n \) is the length of sequence which must be equal to or larger than 4.

On the basis of the initial sequence \( x^{(0)} \), a new sequence

\[ x^{(1)} = (x^{(1)}(1), x^{(1)}(2), \ldots, x^{(1)}(n)) \]

is set up through the accumulated generating operator (AGO), which is monotonically increasing to weaken the variation tendency defined as

\[ x^{(1)}(k) = \sum_{i=1}^{k} x^{(0)}(i). \]

Grey system theory is applied to accumulate generation of \( x^{(0)} \) to obtain a new sequence \( x^{(1)} \), which has a clear growing tendency.

**Step 2.** Establishing the first-order differential equation of grey model GM(1,1) as

\[ \frac{dx^{(1)}}{dt} + ax^{(1)} = b \]

and its difference equation is

\[ x^{(0)}(k) + az^{(1)}(k) = b, \quad 2 \leq k \leq n, \]
where \( a \) is the development coefficient, \( b \) is the driving coefficient, and \( z^{(1)} = (z^{(1)}(2), z^{(1)}(3), \ldots, z^{(1)}(n)) \) is the generated sequence of \( z^{(1)}(k) = ax^{(1)}(k) + (1 - \alpha)x^{(1)}(k - 1) \).

In the original GM(1,1), \( \alpha \) is set to the mean value of adjacent data \( z^{(1)}(k) = 0.5 \cdot x^{(1)}(k) + 0.5 \cdot x^{(1)}(k - 1) \). In this paper, we proposed a method by using the PSO algorithm to find a more efficient value of \( \alpha \).

**Step 3.** From (5), we can obtain the following equation:

\[
\begin{align*}
x^{(0)}(2) + az^{(1)}(2) &= b, \\
x^{(0)}(3) + az^{(1)}(3) &= b, \\
& \vdots \\
x^{(0)}(n) + az^{(1)}(n) &= b.
\end{align*}
\]

From this study, we can find that for the trend prediction of economic data is partially irregular, that the optimization problem be differentiable. Since the authors in [29] found that GM(1,1) model often performs very poor and makes delay errors for quick growth sequences because of the mean value on the generated sequence \( z^{(1)} \). Tan proposed a method that set \( \alpha \) to \((p - 1)/2p\), where \( p = \left(\sum_{k=0}^{\infty}x^{(1)}(k)x^{(1)}(k - 1)\right) \), in order to widen the adaptability of GM(1,1) model to various kinds of time sequences. The authors in [26] found that the RM-GM with variable \( \alpha \) value generates better forecasts than with a fixed \( \alpha \) value. They determined the \( \alpha \) value by the timely percent change. From this study, we can find that for the trend prediction of nonmonotonous functions, the forecast outcomes are much better if the value of \( \alpha \) is set appropriately on the grey predicted results. However, Tan’s method used the whole data set to calculate a fixed value of \( \alpha \). It did not consider the influence of recent data which would improve accuracy.

In an improved RM-GM(1,1) algorithm, the strategy of finding a value of \( \alpha \) could be proposed in a variety of ways. The basic RM-GM(1,1) sets the value of \( \alpha \) to 0.5, which does not consider any influence of sequence data. Although Tan’s strategy could adapt to various sequences, it did not consider the impact of the recent data from the sequence. Chang’s strategy only considered the timely percent change for the prediction. It could perform well on regular and noiseless sequence, but it could not acclimatize itself to the noisy data sequence. In this paper, we select PSO as our strategy to find the value of \( \alpha \) in each loop in \( \alpha \)-RM-GM(1,1). We named our PSO-based algorithm as PRGM(1,1).

### 3.1. Characteristics of PSO

Two significant reasons that make using PSO to calculate the parameter \( \alpha \) are its routinely delivering good optimization results and its simplicity. Compared with another commonly used swarm intelligence method, ant colony optimization (ACO), which is not easy to be used to define variables for the given problems, PSO is not only a metaheuristic that makes few or no assumptions about the problem being optimized, but can also search very large spaces of candidate solutions. It does not require that the optimization problem be differentiable. Since the problem of predicting economic data is partially irregular, noisy, and, changing over time, PSO is a better choice to be employed to optimize parameter \( \alpha \). Another one of the most significant advantages of PSO algorithm is its relatively simple coding and low computational cost. Compared with other optimization algorithms, like ACO, which requires massive computation, PSO can get better results in a faster and
cheaper way [30]. Hence, PSO algorithm can even perform well in the applications that need power-aware computing on smart or personal devices that have limited computational, storage, and energy resources in the case of guarantying the prediction accuracy.

3.2. Calculating a by PSO. The PSO is a population-based optimization technique in which the optimal solution can be found by iteration and the solution quality is evaluated by the fitness. In the PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles. Each particle keeps track of its coordinates in the problem space which are associated with the best solution (fitness) that it has achieved so far. First, a dimensional space $D$ with $m$ particles is initialized. The particles’ position and velocity are randomly initialized. The position of the $r$th particle is represented as $x_i = (x_{i1}, \ldots, x_{id}, \ldots, x_{iD})$ and its velocity is represented as $v_i = (v_{i1}, \ldots, v_{id}, \ldots, v_{iD})$, where $1 \leq i \leq m$ and $1 \leq d \leq D$. Then the objective function values (forecast errors) of all particles can be computed. Then, the particles are updated iteratively until the termination condition is satisfied. It includes the particles’ own speed and location according to the following two formulas for all particles:

$$v_{id}^{k+1} = w \times v_{id}^k + c_1 \times \text{rand} \times (pBest_{id} - x_{id}^k) + c_2 \times \text{rand} \times (gBest_{id} - x_{id}^k),$$

$$x_{id}^{k+1} = x_{id}^k + v_{id}^{k+1}, \quad i = 1, 2, \ldots, m, \quad d = 1, 2, \ldots, D,$$

where, $pBest_{id}$ and $gBest_{id}$ are determined as the objective function values fitness which should be set according to the actual problem solving. For the prediction, it can be set to the smallest prediction error. $pBest_{id}$ and $gBest_{id}$, respectively, represent the individual extreme value of the $r$th particle found by the particle itself at $d$th dimension and the global optimal value which records the best particle among all the particles in the group; $k$ is the pointer of iterations; $c_1$ and $c_2$ are two positive acceleration constants; rand() is the uniform random value in the range $[0, 1]$; $v_{id}^k$ is the velocity of a particle $i$ at iteration $k$; $v_{d\text{min}}^{k} \leq v_{id}^k \leq v_{d\text{max}}$ is the current position of the $r$th particle at iteration $k$; $w$ ($0 \leq w \leq 1$) is the inertia weight determining how much of the particle’s previous velocity is preserved. If the current value is better (with smaller forecast accuracy index value), then update the best position and its objective function value of the particle with the current position and corresponding objective function value. Finally, determine the best particle of the whole population based on their best objective function values. If their objectives function value is smaller than the current global optimal objective function value $\text{Fitness}_{\text{best}}$, then update the best position and objective function value for the entire swarm with the current best particle’s position and objective function value.

3.3. Parameter Selection. In $\alpha$-PSO algorithm, the values for the cognitive weight ($c_1$), social weight ($c_2$), and the inertia weight ($\omega$) having to be selected would have an impact on the convergence speed and the ability of the algorithm to find the optimum. However, different values may be better for different problems. Many works have been done to select a combination of values that works well in a wide range of problems. Both theoretical and empirical studies are available to help in selection of proper values [31–34].

Generally, the individual and sociality weights $c_1$ and $c_2$ are both set to 2. A proper value of inertia weight provides a balance between global and local explorations. A large inertia weight favors global search, while a small inertia weight favors local search [31, 35]. In practice, $\omega$ often reduces linearly from about 0.4 ($\omega_{\text{max}}$) to 0.9 ($\omega_{\text{min}}$). The authors in [31] suggested that utilizing LDW (linear decreasing weight) policy which improved a lot compared with optimization of the benchmark equation algorithm, but not the most common and suitable for the reason that demanding the searching process is linear. It is suggested that for each iteration setting the inertia weight according to the following equation may be a better choice:

$$c_1 (k) = c_1^* - (c_1^* - c_1) \frac{k}{k_{\text{max}}},$$

$$c_2 (k) = c_2^* + (c_2 - c_2^*) \frac{k}{k_{\text{max}}}.$$

A proper value of the inertia weight provides a balance between global and local explorations. A large inertia weight favors global search, while a small inertia weight favors local search. In general, settings near 1 facilitate global search, and settings ranging from $[0.2, 0.5]$ facilitate rapid local search. The linear decreasing weight (16) is introduced to dynamically adapt the inertia weight (13). $w^*$ and $w^\sim$ are usually set to 0.9 and 0.4:

$$w (k) = w^* - (w^* - w^\sim) \frac{k}{k_{\text{max}}}.$$

The nonlinearily decreasing inertia weight (14) incorporates the hyperbolic tangent function (15) to update $w_i$ of each particle $i$:

$$w_i (k) = \frac{1}{1 + \tanh \left( N_{i\text{th}} (k) \right)},$$

$$\tanh (z) = \frac{e^z - e^{-z}}{e^z + e^{-z}},$$

where $N_{ith}$ is the neighborhood index of the particle $i$, which is calculated at each iteration as

$$N_{ith} = \frac{\text{Fitness}^k - gWorst}{gBest - gWorst},$$

where $gWorst^k$ is the global worst fitness value at the current iteration. A small $N_{ith}$ indicates the current position is bad and needs global exploration with a large inertia weight. On the contrary, a large $N_{ith}$ indicates the requirement of local exploitation with a small inertia weight.
The constriction factor $\chi$ was used to control the magnitude of the velocities, instead of $\omega$. The velocity update scheme is replaced with the following:

$$v_{i}^{k+1} = \chi \times \left[v_{i}^{k} + c_1 \times \text{rand} \times \left( p \text{Best}_{i} - a_{i}^{k} \right) \right]$$

$$+ c_2 \times \text{rand} \times \left( g \text{Best} - a_{i}^{k} \right),$$

$$\chi = \frac{2k}{2 - \phi - \sqrt{\phi^2 - 4\phi}},$$

where $\phi = c_1 + c_2$ and generally $k = 1$.

### 4. Experiments and Evaluations

#### 4.1. Datasets

The prediction of the development of tertiary industry is a very important topic in economic and financial areas. However, time series prediction in economic area is generally very difficult because it is nonstationary, nonlinear, and highly noisy.

In order to illustrate that our PRGM(1,1) algorithm gets better performance on both smoothing and noisy data forecasting model by using small set of training data, we used three datasets: financial intermediation in Beijing during 1994 to 2010 which has relatively smoothing trends, real estate in Beijing during 1994 to 2010 which seems much nonlinear, and semiconductor industry production in Taiwan from 1994 to 2002 which seems regular from 1994 to 2000 but irregular since 2000. All datasets are collected from the China Statistical Yearbook, National Bureau of Statistics of China.

#### 4.2. Evaluation Metrics

Prediction accuracy is an important criterion for evaluating a forecasting technique [36]. In this paper, three metrics, namely, mean absolute percentage error (MAPE), mean absolute deviation (MAD), and mean squared error (MSE), which are often adopted for the performance of each model [6, 22], are used to evaluate the prediction accuracy. MAPE is a general accepted metric in percent of prediction accuracy. The criterion of MAPE [37] is listed in Table 1:

<table>
<thead>
<tr>
<th>MAPE (%)</th>
<th>Forecasting power</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;10</td>
<td>Excellent</td>
</tr>
<tr>
<td>10–20</td>
<td>Good</td>
</tr>
<tr>
<td>20–50</td>
<td>Reasonable</td>
</tr>
<tr>
<td>&gt;50</td>
<td>Incorrect</td>
</tr>
</tbody>
</table>

Table 1: Criterion of MAPE.

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{x^{(0)}(i) - x^{(0)}(\hat{i})}{x^{(0)}(i)} \right|.$$  \hspace{1cm} (19)

MAD and MSE are two metrics of the average magnitude of the forecast errors, but the latter imposes a greater penalty on a large error than several small errors. The smaller the values, the closer the predicted values to the actual values [38]:

$$\text{MAD} = \frac{1}{n} \sum_{i=1}^{n} \left| x^{(0)}(i) - x^{(0)}(\hat{i}) \right|.$$  \hspace{1cm} (20)

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} \left[ x^{(0)}(i) - x^{(0)}(\hat{i}) \right]^2.$$  \hspace{1cm} (21)

Besides, the coefficient of determination, denoted as $r^2$, is also applied to evaluate models in our experiments:

$$r^2 = 1 - \frac{\text{SSE}}{\text{SST}}.$$  \hspace{1cm} (21)

where

$$\text{SSE} = \sum_{k=1}^{n} \left( x^{(0)}(k) - x^{(0)}(\hat{k}) \right)^2, \ k = 1, 2, \ldots, n,$$

$$\text{SST} = \left( \sum_{k=1}^{n} x^{(0)}(k) \right)^2 / n, \ k = 1, 2, \ldots, n.$$

The higher the value of $r^2$ is, the more successful the model is at predicting statistical data [23]. The maximum value of the coefficient of determination $r^2$ is 1.

#### 4.3. Experimental Setup

The experiments are divided into two parts, Experiment I and Experiment II. Experiment I used the datasets of financial intermediation and real estate in Beijing. The data from 1994 to 2005 were used as sample data, while the data from 2006 to 2010 were used for prediction and test. Experiment I compared three prediction models on these data, GM(1,1), RM-GM(1,1), and PRGM(1,1). Experiment II compared various PRGM(1,1) with different parameter settings.

The values of the parameters for PRGM(1,1) are selected in both experiments. We set the number of candidates of $\alpha$ in particle searching space partimum to 1,000 and the maximum number of iterations $\text{iter}_{\text{max}}$ to 100. For the basic PRGM(1,1), we set the two weights, $c_1 = 2$ and $c_2 = 2$.

#### 4.4. Experiment I

Table 2 shows the parameters calculated by the three prediction models, GM(1,1), RM-GM(1,1), and PRGM(1,1). In GM(1,1) which is constructed by all of the data 1994–2005 with the fixed $\alpha$ value 0.5, the parameter $a$ is equal to a fixed value $-0.148$ and $b$ is also equal to a fixed value 165.061 for all the predicted years in financial intermediation. Similarly, $a = -0.254$ and $b = 42.152$ for all the predicted years in real estate.

In RM-GM(1,1), we set the sample sequence with $l = 12$ and $m = 1$ starting from 1994 to forecast the data from 2006 to 2010. Hence, the rolling number $k$ equals 5. The $\alpha$ value is also fixed to 0.5 in RM-GM(1,1). However, the parameters $a$ and $b$ change for every predicted year because of the rolling mechanism.

In PRGM(1,1), similar with RM-GM(1,1), the sample sequence with $l = 12$ and $m = 1$ that starts from 1994 to 2005 was used for predicting the 5 years’ data since 2006. However, the value of $\alpha$ is a variable of year that is different among the predictions of 2006–2010. Hence, the parameters $a$ and
Table 2: The parameter values calculated by GM(1, 1), RM-GM(1, 1), and PRGM(1, 1), respectively.

<table>
<thead>
<tr>
<th>Year</th>
<th>GM(1, 1)</th>
<th>RM-GM(1, 1)</th>
<th>PRGM(1, 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>α</td>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>Financial</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>intermediation</td>
<td>2006</td>
<td>−0.148</td>
<td>165.061</td>
</tr>
<tr>
<td></td>
<td>2008</td>
<td>0.500</td>
<td>−0.148</td>
</tr>
<tr>
<td></td>
<td>2009</td>
<td>−0.143</td>
<td>251.047</td>
</tr>
<tr>
<td>Real estate</td>
<td>2006</td>
<td>−0.254</td>
<td>42.152</td>
</tr>
<tr>
<td></td>
<td>2008</td>
<td>0.500</td>
<td>−0.254</td>
</tr>
<tr>
<td></td>
<td>2009</td>
<td>−0.285</td>
<td>41.182</td>
</tr>
<tr>
<td></td>
<td></td>
<td>−0.273</td>
<td>78.809</td>
</tr>
</tbody>
</table>

Table 3: The evaluation metrics to compare GM(1, 1), RM-GM(1, 1), and PRGM(1, 1).

<table>
<thead>
<tr>
<th></th>
<th>GM(1, 1)</th>
<th>RM-GM(1, 1)</th>
<th>PRGM(1, 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Financial intermediation</td>
<td>MAPE (%)</td>
<td>6.3452</td>
<td>8.3619</td>
</tr>
<tr>
<td></td>
<td>MAD</td>
<td>93.1390</td>
<td>126.4600</td>
</tr>
<tr>
<td></td>
<td>MSE</td>
<td>12666.000</td>
<td>20607.000</td>
</tr>
<tr>
<td></td>
<td>r²</td>
<td>0.8560</td>
<td>0.7657</td>
</tr>
<tr>
<td>Real estate</td>
<td>MAPE (%)</td>
<td>63.8925</td>
<td>61.7673</td>
</tr>
<tr>
<td></td>
<td>MAD</td>
<td>599.65</td>
<td>579.980</td>
</tr>
<tr>
<td></td>
<td>MSE</td>
<td>520120.00</td>
<td>494280.000</td>
</tr>
<tr>
<td></td>
<td>r²</td>
<td>−24.2190</td>
<td>−22.9660</td>
</tr>
</tbody>
</table>

4.5. Experiment II. In this experiment, we estimated PSO variants of different parameter configurations. We evaluated the constant setting and linearly varying settings of $c_1$ and $c_2$ on prediction accuracy. In constant settings, the configuration of $c_1 = c_2 = 1.5$ is the best. It is in accordance with most of the previous conclusions. In linearly varying setting (see (12)), there is not much improvement on the metrics compared with the constant setting. We also evaluated the forecasting performances with diverse combinations of the start values $c_1^0$ and $c_2^0$, and the end values $c_1^e$ and $c_2^e$ ranging from $[0.5, 4]$ with a step of 0.5 and found that there still is not much difference among them for all of the three datasets. We evaluated three kinds of $w$ settings, constant, linearly decreasing, and nonlinearly decreasing. In constant setting, the optimal setting is $w = 0.5$ for all of the datasets. We also observed that the performance is exactly the same when...
Table 4: The forecasting data and evaluation metrics produced by PRGM(1, 1) and \( P \) value RM-GM(1, 1).

<table>
<thead>
<tr>
<th>Year</th>
<th>Actual value</th>
<th>( P ) value RM-GM(1, 1)</th>
<th>PRGM(1, 1)</th>
<th>PRGM(1, 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Predictive value</td>
<td>Error</td>
<td>Predictive value</td>
</tr>
<tr>
<td>Semiconductor industry production</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1998</td>
<td>2834</td>
<td>3020</td>
<td>6.56</td>
<td>2830</td>
</tr>
<tr>
<td>1999</td>
<td>4235</td>
<td>3749</td>
<td>11.48</td>
<td>4096</td>
</tr>
<tr>
<td>2000</td>
<td>7144</td>
<td>6546</td>
<td>8.37</td>
<td>6792</td>
</tr>
<tr>
<td>2001</td>
<td>5269</td>
<td>6624</td>
<td>25.72</td>
<td>6412</td>
</tr>
<tr>
<td>2002</td>
<td>6529</td>
<td>6498</td>
<td>0.48</td>
<td>7302</td>
</tr>
<tr>
<td>MAPE (%)</td>
<td></td>
<td>10.52</td>
<td></td>
<td>8.3787</td>
</tr>
</tbody>
</table>

Figure 1: The influence of PSOs with various parameter settings on MAPE and \( r^2 \).

Figure 2 shows an illustration of the evolution of the fitness at the first predicted year in all of the datasets. According to our empirical study, the maximum iteration \( k_{\text{max}} \) can be set to 60–80 in the single particle PSO. Figure 2 shows the comparison of the convergence speed among variant PSOs. There is no general rule on these PSOs for all of the datasets, but all PSOs converge after 60–80 iterations at most. The time complexity of the PSO is \( O(k_{\text{max}} \cdot m \cdot O(\text{Fitness})) \). The runtime is dependent on both population size and iteration number.

5. Conclusions

In this paper, we proposed a rolling mechanism based grey model, and its parameter \( \alpha \) is optimized by the PSO algorithm, which has the significant impact of the forecast accuracy. The experiments show that the prediction made by
PRGM(1,1) model is almost perfect among three economic datasets, which are either regular or noisy. PRGM(1,1) gets much better forecast accuracy compared with three widely used grey models: GM(1,1) that has a fixed \( \alpha \) and ignores the impact of recent data, RM-GM(1,1) that considers the impact of recent data but has a fixed \( \alpha \) through the whole prediction period, and \( P \) value RM-GM(1,1) that not only considers the recent data but also adjusts \( \alpha \) in each rolling step.

We evaluated other variant PSOs with different parameter settings. Almost all of metaheuristics are required to set a number of parameters, which might lead to different outcomes, for example, multiple locally optimal solutions in the parameter space in terms of solution quality. An extension of this work includes analyzing the principles of balancing exploitation and exploration of metaheuristics on forecasting. We will focus on the work of the details of comparing the effectiveness of the exploitation or the exploration among
them and analyzing the different concepts or philosophy within them.

Conflict of Interests
The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments
This work is supported by Cuiying Grant of China Telecom, Gansu Branch (Grant no. Izudxy-2013-3), Science and Technology Planning Project of Chengguan District, Lanzhou (Grant no. 2013-3-1), and Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry (Grant no. 44th). This work is also partially supported by Fundamental Research Funds for the Central Universities (Grant no. XDJK2014C141 and SWU114005).

References


