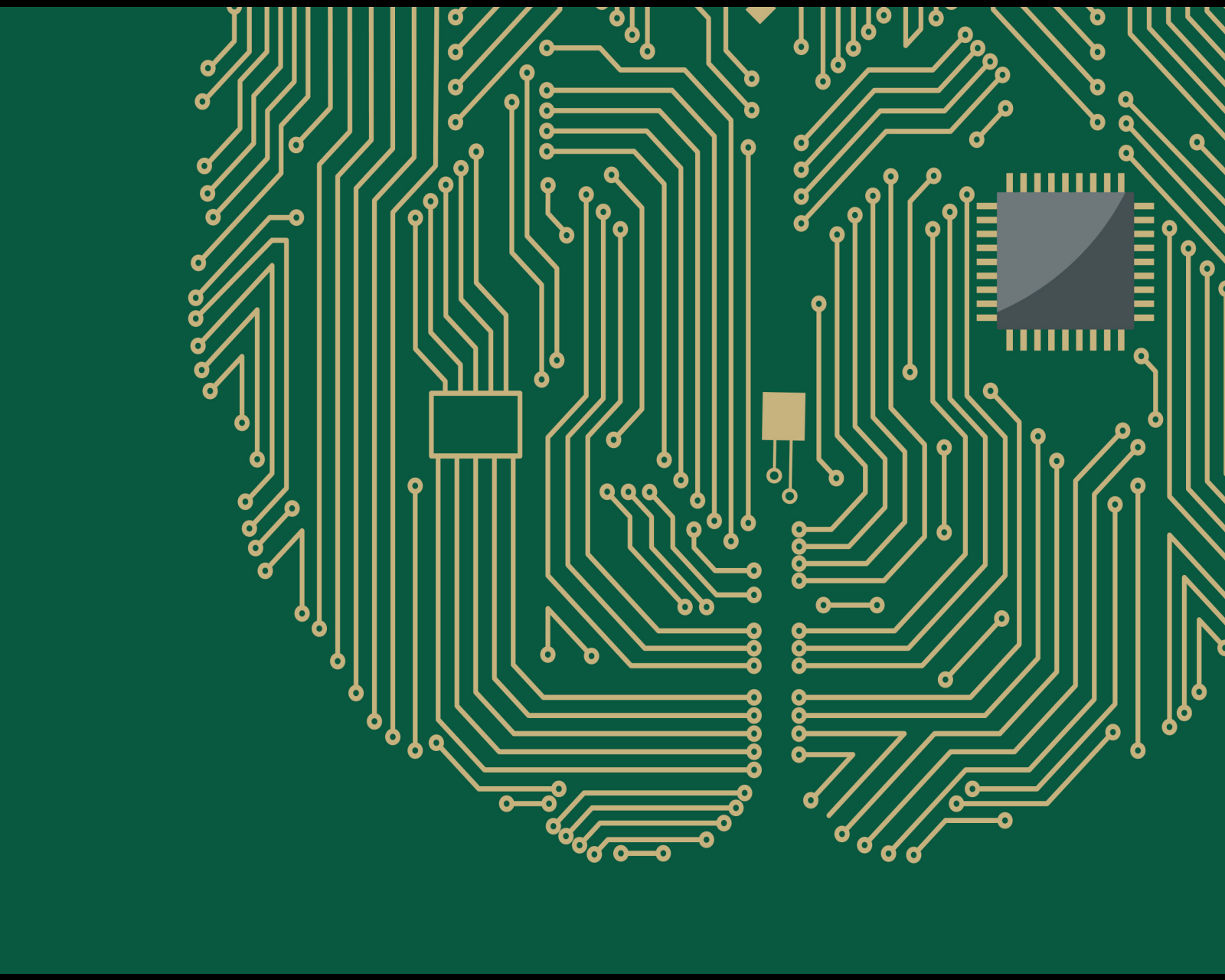
Computational Intelligence and Neuroscience

Fusion of Computational Intelligence Techniques and Their Practical Applications

Guest Editors: Rahib H. Abiyev, Rafik Aliyev, Okyay Kaynak, I. Burhan Turksen, and Karl Walter Bonfig



Fusion of Computational Intelligence Techniques and Their Practical Applications

Computational Intelligence and Neuroscience

Fusion of Computational Intelligence Techniques and Their Practical Applications

Guest Editors: Rahib H. Abiyev, Rafik Aliyev, Okyay Kaynak, I. Burhan Turksen, and Karl Walter Bonfig

Editorial Board

Contents

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 463147, 3 pages
http://dx.doi.org/10.1155/2015/463147

Editorial

Fusion of Computational Intelligence Techniques and Their Practical Applications

Rahib H. Abiyev,[1] Rafik Aliev,[2] Okyay Kaynak,[3] I. Burhan Turksen,[4] and Karl Walter Bonfig[5]

[1]*Faculty of Engineering, Applied Artificial Intelligence Research Centre Near East University, Lefkoşa, North Cyprus, Mersın 10, Turkey*
[2]*Department of Computer-Aided Control System, Azerbaijan State Oil Academy, Azadlik Avenue 20, Baku, Azerbaijan*
[3]*Department of Electrical and Electronic Engineering, Bogazici University, Bebek, 80815 Istanbul, Turkey*
[4]*University of Toronto, 170 College Street (Rear), Haultain Building, Room 305A, Toronto, ON, Canada M5S 3G8*
[5]*Department of Electrical Engineering and Information, University of Siegen, 57068 Siegen, Germany*

Correspondence should be addressed to Rahib H. Abiyev; rahib.abiyev@neu.edu.tr

Received 22 June 2015; Accepted 22 June 2015

Computational intelligence techniques inspired by evolution, by nature, and by the brain are playing an important role in the solution of complex real-world problems. Fusion of computational intelligence techniques integrates neural networks, fuzzy systems, and evolutionary computing into a system design that enables handling of complexity and managing of uncertainty and imprecision. Each respective technique enhances the capability of the other and the fusion of these paradigms in system design offsets the demerits of one paradigm by the merits of another. Recently, computational intelligence techniques have been widely applied to a wide variety of complex problems, including engineering, science, and business. However, due to complexity and uncertainty in these problems, it becomes difficult to find out the optimal solution of the problems. Hereby it is necessary to consider the latest trends and developments in the field of fusion of computational intelligence techniques and to develop efficient computational models for solving practical problems. Fusion of computational intelligence techniques covers the spectrum of applications, comprehensively demonstrating the advantages of fusion techniques in industrial applications that deal with various kinds of inaccuracies and uncertainties.

The aim of this special issue was the presentation of research articles as well as review articles incorporating the contributions in theories, the structures, algorithms, and advances in the design of system based on fusion of neural networks, fuzzy systems, and evolutionary algorithms and their practical applications.

The following is a brief summary for each of the accepted articles.

The paper "A Method for Estimating View Transformations from Image Correspondences Based on the Harmony Search Algorithm" by E. Cuevas and M. Díaz presents a new improved algorithm that combines the random sampling consensus (RANSAC) method and the Harmony Search (HS) algorithm for estimation of the model parameters from a data set. The proposed algorithm adopts a different sampling strategy than RANSAC to generate putative solutions and can substantially reduce the number of learning iterations. At each iteration, new candidate solutions are generated iteratively by taking into consideration the quality of models produced by previous candidate solutions. The rules for the generation of candidate solutions are motivated by the improvisation process that occurs when a musician searches for a better state of harmony. The method is used for the estimation of homographies, considering synthetic and real images, and it is also employed for position estimation in a humanoid robot.

The paper "Application of Z-Number Based Modeling in Psychological Research" by R. Aliev and K. Memmedova uses Z-number based fuzzy approach for modelling the effect of Pilates exercises on motivation, attention, anxiety, and educational achievement of students. The grade point average of the students was used as the measure of educational achievement. The inference techniques for approximate reasoning based on Z-interpolation method suggested by Zadeh are used in the

decision making process. The basic steps of Z-number based modelling with numerical solutions are presented.

The paper "Fusing Swarm Intelligence and Self-Assembly for Optimizing Echo State Networks" by C. E. Martin and J. A. Reggia considers the optimizing of a neural network's weights and topology using integration of self-assembly (SA) and particle swarm optimization (PSO). The authors developed a model that integrates network self-assembly and particle swarm optimization for the purpose of growing neural networks with weights and topologies that are optimized for specified computational tasks. The presented model is used for optimizing echo state network weights and topologies on a number of challenging benchmark problems from the domains of time-series forecasting and control.

The paper "Cuckoo Search Algorithm Based on Repeat-Cycle Asymptotic Self-Learning and Self-Evolving Disturbance for Function Optimization" by J. Wang et al. developed a new improved cuckoo search algorithm based on the repeat-cycle asymptotic self-learning and self-evolving disturbance (RC-SSCS). A disturbance operation is added to the algorithm in order to make a more careful search near the bird's nests location. The repeat-cycle asymptotic mode is to narrow the disturbance scope based on the last disturbance results and then go on the next disturbance. The proposed algorithm improves convergence velocity and optimization accuracy of the cuckoo search (CS) algorithm for solving the function optimization problems. This improved algorithm overcomes the CS algorithm's defects which result from its high degree of random and strong leap and also makes full use of the information near the bird's nest location that had been found. The comparative results with the benchmarking functions show that the improved cuckoo search algorithm has better convergence velocity and optimization accuracy.

The paper "Symmetry Based Automatic Evolution of Clusters: A New Approach to Data Clustering" by S. Vijendra and S. Laxman presents a multiobjective genetic clustering approach, in which data points are assigned to clusters based on new line symmetry distance. The proposed multiobjective line symmetry based genetic clustering (MOLGC) algorithm evolves near-optimal clustering solutions using multiple clustering criteria, without a priori knowledge of the actual number of clusters. The multiple randomized dimensional trees based nearest neighbour search is used to reduce the complexity of finding the closest symmetric points. Experimental results show that proposed algorithm can obtain optimal clustering solutions in terms of different cluster quality measures.

The paper "Optimization of High-Dimensional Functions through Hypercube Evaluation" by R. H. Abiyev and M. Tunay proposes a novel evolutionary learning algorithm based on evaluation and optimization of a hypercube for solving global numerical optimization problems. The algorithm is inspired from the behaviour of doves discovering new areas for food in natural life. The HO algorithm comprises the initialization and evaluation process, displacement-shrink process, and searching space process. The initialization and evaluation process initializes initial solution and evaluates the solutions in given hypercube. The displacement-shrink process determines displacement and evaluates objective functions using new points; the search area process determines next hypercube using certain rules and evaluates the new solutions. The HO algorithm is tested on a set of specific benchmarking functions and has shown better performance for global optimization of both low- and high-dimensional problems with large numbers of local optimal.

The paper "A Simple Fitness Function for Minimum Attribute Reduction" by Y. Su et al. considers the problem of finding the minimal subset of the condition attribute set such that minimal set has the same classification quality as the condition attribute set. For this purpose, the design of fitness function that satisfies the equivalence between the optimal solution and the minimal attribute reduction is considered. The optimality and adequacy of the fitness function were tested experimentally. Experimental results show that the proposed fitness function is better than existing fitness functions for each algorithm used in the test.

The paper "Expected Utility Based Decision Making under Z-Information and Its Application" by R. R. Aliev et al. presents decision making under Z-information based on direct computation over Z-numbers. Z-numbers based formalization of information represents a natural language-(NL-) based value of a variable of interest in line with the related NL-based reliability. The approach utilizes expected utility paradigm and is applied to a benchmark decision problem in economics.

The paper "An Intelligent Model for Pairs Trading Using Genetic Algorithms" by C.-F. Huang et al. presents the solution of pairs trading problem using genetic algorithms (GA). In this problem, the pairs of stocks are bought and sold in pair combinations for arbitrage opportunities. The results showed that the GA-based models are generating robust models to tackle the dynamic characteristics in the financial application and provide an effective solution to pairs trading for investment in practice.

The paper "An Enhanced Differential Evolution with Elite Chaotic Local Search" by Z. Guo et al. presents an enhanced differential evolution with elite chaotic local search (DEECL) to solve complex optimization problems. The algorithm utilizes a chaotic search strategy based on the heuristic information from the elite individuals to promote the exploitation power. The experimental results using classical test functions show that DEECL is very competitive on the majority of the test functions.

The paper "A Multiuser Manufacturing Resource Service Composition Method Based on the Bees Algorithm" by Y. Xie et al. presents multiuser resource service composition (RSC) for modelling of an optimal resource service allocation in current open and service-oriented manufacturing model. The model takes into account both subjective and objective quality of service properties as representatives to evaluate a solution. The basic Bees Algorithm is tailored for finding a near-optimal solution to the model. Particular rules are designed for handling the constraints and finding Pareto optimality. In addition, the established model introduces a trusted service set to each user so that the algorithm could start by searching in the neighbour of more reliable service chains (known as seeds) than those randomly generated.

The paper "Phase Response Design of Recursive All-Pass Digital Filters Using a Modified PSO Algorithm" by W.-D. Chang presents a design scheme for the phase response of an all-pass recursive digital filter using a modified PSO (MPSO) algorithm. In the MPSO algorithm, a new adjusting factor is introduced in the velocity updating formula in order to improve the searching ability. The algorithm is applied to design the coefficients of the filter. Two different kinds of linear phase response design examples are illustrated. The obtained results show that the MPSO is superior to the general PSO for the phase response design of the digital recursive all-pass filter.

The paper "Application of Boosting Regression Trees to Preliminary Cost Estimation in Building Construction Projects" by Y. Shin uses a boosting regression tree (BRT) algorithm for estimating the cost at the early stage of a construction project. The boosting approach has attracted attention because of its effective learning algorithm and strong boundaries in terms of its generalization performance. The used BRT model provides additional information such as the importance plot and structure model, which can support estimators in comprehending the decision making process. Consequently, the boosting approach has shown a high performance in cost estimations in a building construction project.

The paper "Predictive Modeling in Race Walking" by K. Wiktorowicz et al. presents the use of linear and nonlinear multivariable models for prediction sports results of athletes practicing race walking. These models are calculated using data collected from race walkers' training events and they are used to predict the result over a 3 km race based on training loads. The paper proposes the nonlinear modifications for linear models and artificial neural networks in order to achieve smaller prediction error. It was shown that the best model is a modified LASSO regression with quadratic terms in the nonlinear part. This model has the smallest prediction error and simplified structure by eliminating some of the predictors.

The paper "Prognostics of Lithium-Ion Batteries Based on Wavelet Denoising and DE-RVM" by C. Zhang et al. presents a novel battery capacity prognostics approach in order to estimate the remaining useful life (RUL) of lithium-ion batteries. Wavelet denoising is performed twice with different thresholds in order to weaken the strong noise and remove the weak noise. Relevance vector machine (RVM) improved by differential evolution (DE) algorithm is utilized to estimate the battery RUL based on the denoised data. An experiment involving battery 5 capacity prognostics case and battery 18 capacity prognostics case is conducted and validated that the proposed approach can predict the trend of battery capacity trajectory closely and estimate the RUL precisely.

The paper "Incremental Discriminant Analysis in Tensor Space" by L. Chang et al. presents a machine learning algorithm in tensor space. The algorithm employs tensor representation to carry on discriminant analysis and combine incremental learning to alleviate the computational cost. The experiments on facial image detection have shown that the algorithm improves the performance of the system compared with other algorithms and reduces the computational issues apparently.

Although the above papers do not completely cover all the aspects of fusion of computational intelligence techniques, they provide important issues and the benefits of practical applications of computational intelligence techniques in engineering and science.

Acknowledgments

We would like to thank all the authors for submitting their papers to the special issue as well as the reviewers for providing their expertise and making valuable comments.

Rahib H. Abiyev
Rafik Aliev
Okyay Kaynak
I. Burhan Turksen
Karl Walter Bonfig

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 918305, 8 pages
http://dx.doi.org/10.1155/2015/918305

Research Article

Prognostics of Lithium-Ion Batteries Based on Wavelet Denoising and DE-RVM

Chaolong Zhang,[1,2] Yigang He,[1] Lifeng Yuan,[1] Sheng Xiang,[1] and Jinping Wang[1]

[1] *School of Electrical Engineering and Automation, Hefei University of Technology, Hefei 230009, China*
[2] *School of Physics and Electronic Engineering, Anqing Normal University, Anqing 246011, China*

Correspondence should be addressed to Chaolong Zhang; zhangchaolong@126.com

Received 8 December 2014; Accepted 10 May 2015

Academic Editor: Rafik Aliyev

Lithium-ion batteries are widely used in many electronic systems. Therefore, it is significantly important to estimate the lithium-ion battery's remaining useful life (RUL), yet very difficult. One important reason is that the measured battery capacity data are often subject to the different levels of noise pollution. In this paper, a novel battery capacity prognostics approach is presented to estimate the RUL of lithium-ion batteries. Wavelet denoising is performed with different thresholds in order to weaken the strong noise and remove the weak noise. Relevance vector machine (RVM) improved by differential evolution (DE) algorithm is utilized to estimate the battery RUL based on the denoised data. An experiment including battery 5 capacity prognostics case and battery 18 capacity prognostics case is conducted and validated that the proposed approach can predict the trend of battery capacity trajectory closely and estimate the battery RUL accurately.

1. Introduction

Lithium-ion batteries have been widely used as crucial components and important backup elements for many systems including electric vehicles, consumer electronics, and aerospace electronics. Compared with other kind of batteries, the lithium-ion battery has advantages of high power density, high galvanic potential, light weight, and long cycle life. However, irreversible chemical and physical changes take place in the lithium-ion battery with usage and aging. As a result, the battery health degrades gradually until it is no longer usable eventually. The consequence of the battery failure would lead to the capacity degradation, operation loss, downtime, and even catastrophic failure. Hence, prognostics and health management (PHM) of the lithium-ion battery has been an active field which has attracted an increasing attention today [1–12].

PHM is an enabling discipline composed of technologies and approaches to estimate the reliability of an application system in its actual life cycle conditions to provide ample forewarning before a failure occurs and mitigates system risk. PHM of the lithium-ion battery includes evaluating its state of health (SOH) and predicting its remaining useful life (RUL). Meanwhile, the gradual decreased capacity of the battery is a universally used SOH indicator that can track its health degradation.

Model-based and data-driven approaches are two main kinds of approaches to the battery capacity prognostics. Model-based approaches employ mathematical representations to character the understanding of the battery failure and underlying the battery capacity's degradation model. Extended Kalman filtering (EKF) [1, 2], nonlinear model [3, 4], and particle filtering (PF) [5, 6] are commonly used model-based methods for the battery capacity estimation. However, an accurately analytical and universally accepted model to track the battery capacity degradation and evaluate the battery RUL is usually difficult to be derived because of the complex electronic system, noise, data availability, uncertain environments, and application constraints. Data-driven approaches utilize statistical and machine learning techniques to evaluate the battery capacity and predict the battery RUL. The approaches avoid constructing complex

physical models and have been applied in many relative works [7–12]. Artificial neural network is a widely used data-driven approach to the battery capacity prognostics [7, 8]. However, it has disadvantages of poor generalization, difficult structure confirmation, and low convergence rate. Support vector machine (SVM) is a machine learning tool [13] characterized by the usage of kernel functions and it has been utilized to estimate the battery RUL [9, 10]. Relevance vector machine (RVM) is a Bayesian sparse kernel technique [14] with usage of much fewer kernel functions and higher performance compared to the SVM. Meanwhile, RVM has been applied to the research field [11, 12].

The measured battery capacity data are often subject to the different levels of noise pollution because of the impact of disturbances, measurement errors, stochastic load, and other unknown behaviors in batteries. Capacity prognostics based on the noisy data cannot produce accurate predict results. Therefore, it is significantly important to preprocess the measured capacity data for the purpose of extracting the original data and removing the noise. To address the problem and estimate the battery RUL accurately, a novel battery capacity prognostics approach is presented in the paper. A wavelet denoising method performed with different thresholds is employed to process the measured data to reduce the uncertainty and extract the useful information. RVM optimized by differential evolution (DE) algorithm is utilized to estimate the battery RUL. An experiment including battery 5 capacity prognostics case and battery 18 capacity prognostics case is conducted, which validates that the proposed approach can predict the trend of battery capacity trajectory closely and estimate the RUL accurately.

The material in the paper is organized in the following order: Section 2 describes the strategy of wavelet denoising method. Section 3 introduces RVM algorithm and its parameter optimization by using DE algorithm. Section 4 illustrates the experiment procedure, presents the experiment results, and gives discussions. Finally, conclusions are drawn in Section 5.

2. Wavelet Denoising

The measured capacity data of batteries often suffer from the different levels of noise pollution. Experiment with noisy data cannot yield the accurate RUL. As a result, it is very important to preprocess the capacity data for the purpose of extracting the original data. Wavelet denoising method is adopted to address the concern.

Assume that the measured capacity data capacity(c) is comprised by

$$\text{capacity}\,(c) = x\,(c) + \sigma\,(c)\,, \tag{1}$$

where $x(c)$ is the original data; $\sigma(c)$ is the additive noise; and c refers to the cycle which is a time index.

Assume that Z is an integers set, $\{V_t\}_{t\in Z}$ is an orthogonal multiresolution analysis, and $\{W_t\}_{t\in Z}$ is the associated wavelet space. The capacity(c) projection on V_t is

$$P_{V_t} = P_{V_{t+1}} + P_{W_{t+1}} = \sum_{i\in Z} c_{t+1}^{i}\phi_{t+1,i} + \sum_{i\in Z} d_{t+1}^{i}\psi_{t+1,i}, \tag{2}$$

where $P_{V_{t+1}}$ and $P_{W_{t+1}}$ denote the capacity(c) projections on V_{t+1} and W_{t+1} at 2^{t+1} resolution, respectively; c_{t+1}^{i} and d_{t+1}^{i} refer to the scaling coefficient and wavelet coefficient of capacity(c) at 2^{t+1} resolution, respectively; ϕ_{t+1} and ψ_{t+1} represent the scaling function and wavelet function of capacity(c) at 2^{t+1} resolution, respectively. Therefore, c_{t+1} and d_{t+1} characterize the approximations and details of capacity(c) at 2^{t+1} resolution, respectively. Correspondingly, $\{V_t\}_{t\in Z}$ can be decomposed as

$$\begin{aligned} V_t &= W_{t+1} \oplus V_{t+1} = W_{t+1} \oplus \left(W_{t+2} \oplus V_{t+2}\right) \\ &= W_{t+1} \oplus W_{t+2} \oplus \left(W_{t+3} \oplus V_{t+3}\right) \\ &= W_{t+1} \oplus W_{t+2} \oplus W_{t+3} \oplus \cdots . \end{aligned} \tag{3}$$

By using the multilevel wavelet decomposition, discrete approximation coefficients and detail coefficients are produced. The detail coefficients with small absolute values are considered to be noise. Generally, the traditional wavelet denoising method is setting the detail coefficients below a threshold to zero and reconstructing the denoised data by using the rest coefficients. Sqtwolog threshold, rigorous threshold, heursure threshold, and minimax threshold are commonly used rules to yield the threshold. In the work, the wavelet denoising is performed twice with sqtwolog threshold rule and minimax threshold rule, respectively.

The sqtwolog threshold rule produces the threshold which can yield good performance multiplied by a small factor proportional to log(length(capacity)):

$$\text{Threshold}_{\text{sqtwolog}} = \sqrt{2\log n}, \tag{4}$$

where n is the length of the capacity set.

The minimax threshold rule brings about the minimum of the maximum mean square error generated for the worst function with a given set by using the minimax principle. The threshold is defined as

$$\text{Threshold}_{\text{minimax}} = a + b * \frac{\log\,(n)}{\log\,(2)}, \tag{5}$$

where a and b are factors which are generally set to 0.3936 and 0.1829, respectively.

The minimax threshold is obviously lower than the sqtwolog threshold in magnitude with a signal. Wavelet denoising with the sqtwolog threshold can weaken the strong noise obviously. Meanwhile, wavelet denoising with the minimax threshold can remove the weak noise effectively. The wavelet denoising strategy in the work is implementing wavelet denoising with the sqtwolog threshold firstly and then performing wavelet denoising with the minimax threshold.

3. DE-RVM

3.1. RVM. RVM is firstly presented in [14] and has generated demonstrative effect in prognostics [15–17]. The algorithm is a Bayesian treatment which provides probabilistic interpretation of the output. The relevance vectors and weights are obtained by maximizing a marginal likelihood.

Assume that $\{\mathbf{x}_i, t_i\}_{i=1}^N$ is the input data. The target t_i is obtained by

$$t_i = y(\mathbf{x}_i; \mathbf{w}) + \varepsilon_i, \tag{6}$$

where $\mathbf{w} = (w_0, w_1, \ldots, w_N)^T$ and ε_i is the noise with mean zero and variance σ^2.

Assume that t_i is independent and the likelihood of complete dataset can be defined as

$$p(\mathbf{t} \mid \mathbf{w}, \sigma^2) = (2\pi\sigma^2)^{-N/2} \exp\left\{-\frac{1}{2\sigma^2}\|\mathbf{t} - \boldsymbol{\varphi}\mathbf{w}\|^2\right\}, \tag{7}$$

where $\mathbf{t} = (t_1, t_2, \ldots, t_N)^T$ and $\boldsymbol{\varphi}$ is a $N \times (N+1)$ design matrix with $\boldsymbol{\varphi} = [\varphi(\mathbf{x}_1), \varphi(\mathbf{x}_2), \ldots, \varphi(\mathbf{x}_N)]^T$ and $\varphi(\mathbf{x}_i) = [1, K(\mathbf{x}_i, \mathbf{x}_1), K(\mathbf{x}_i, \mathbf{x}_2), \ldots, K(\mathbf{x}_i, \mathbf{x}_N)]^T$.

Maximum likelihood estimations of $\mathbf{w}$ and σ^2 in (7) often result in overfitting. Hence, an explicit zero-mean Gaussian prior probability distribution is defined in order to constrain the parameters as

$$p(\mathbf{w} \mid \boldsymbol{\alpha}) = \prod_{i=0}^{N} N\left(w_i \mid 0, \alpha_i^{-1}\right), \tag{8}$$

where $\boldsymbol{\alpha}$ is a $N+1$ hyperparameters vector.

Using Bayes' rule, the posterior probability about all of the unknown parameters can be obtained by

$$p\left(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2 \mid \mathbf{t}\right) = \frac{p\left(\mathbf{t} \mid \mathbf{w}, \boldsymbol{\alpha}, \sigma^2\right) p\left(\mathbf{w}, \alpha, \sigma^2\right)}{\int p(\mathbf{t} \mid \mathbf{w}, \boldsymbol{\alpha}, \sigma^2)\, p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)\, d\mathbf{w}\, d\boldsymbol{\alpha}\, d\sigma^2}. \tag{9}$$

However, the normalizing integral $\int p(\mathbf{t} \mid \mathbf{w}, \boldsymbol{\alpha}, \sigma^2) p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2) d\mathbf{w}\, d\boldsymbol{\alpha}\, d\sigma^2$ cannot be easily executed. Therefore, $p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2 \mid \mathbf{t})$ can be instead decomposed as

$$p\left(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2 \mid \mathbf{t}\right) = p\left(\mathbf{w} \mid \mathbf{t}, \boldsymbol{\alpha}, \sigma^2\right) p\left(\boldsymbol{\alpha}, \sigma^2 \mid \mathbf{t}\right). \tag{10}$$

Based on the Bayes' rule, the posterior distribution of weights is obtained through

$$\begin{aligned} p\left(\mathbf{w} \mid \mathbf{t}, \boldsymbol{\alpha}, \sigma^2\right) &= \frac{p\left(\mathbf{t} \mid \mathbf{w}, \sigma^2\right) p(\mathbf{w} \mid \boldsymbol{\alpha})}{p\left(\mathbf{t} \mid \boldsymbol{\alpha}, \sigma^2\right)}, \\ p\left(\mathbf{w} \mid \mathbf{t}, \boldsymbol{\alpha}, \sigma^2\right) &= (2\pi)^{-(N+1)/2} |\boldsymbol{\Sigma}|^{-1/2} \\ &\quad \cdot \exp\left\{-\frac{1}{2}(\mathbf{w} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{w} - \boldsymbol{\mu})\right\}, \end{aligned} \tag{11}$$

where the posterior mean and covariance are

$$\begin{aligned} \boldsymbol{\mu} &= \sigma^{-2} \boldsymbol{\Sigma} \boldsymbol{\varphi}^T \mathbf{t}, \\ \boldsymbol{\Sigma} &= \left(\sigma^{-2} \boldsymbol{\varphi}^T \boldsymbol{\varphi} + A\right)^{-1}, \end{aligned} \tag{12}$$

where $A = \mathrm{diag}(\alpha_0, \alpha_1, \ldots, \alpha_N)$.

Because of the uniform hyperpriors, $p(\mathbf{t} \mid \boldsymbol{\alpha}, \sigma^2)$ is described by

$$\begin{aligned} p\left(\mathbf{t} \mid \boldsymbol{\alpha}, \sigma^2\right) &= \int p\left(\mathbf{t} \mid \mathbf{w}, \sigma^2\right) p(\mathbf{w} \mid \boldsymbol{\alpha})\, d\mathbf{w} \\ &= (2\pi)^{-N/2} \left|\sigma^2 \mathbf{I} + \boldsymbol{\varphi} A^{-1} \boldsymbol{\varphi}^T\right|^{-1/2} \\ &\quad \cdot \exp\left\{-\frac{1}{2}\mathbf{t}^T \left(\sigma^2 \mathbf{I} + \boldsymbol{\varphi} A^{-1} \boldsymbol{\varphi}^T\right)^{-1} \mathbf{t}\right\}. \end{aligned} \tag{13}$$

The maximum posterior (MP) estimate of the weights is described by the posterior mean, which depends on the value of $\boldsymbol{\alpha}$ and σ^2. The estimates of $\boldsymbol{\alpha}_{\mathrm{MP}}$ and σ^2_{MP} are acquired by maximizing the marginal likelihood. Tipping [14] presents the iterative formulas for $\boldsymbol{\alpha}_{\mathrm{MP}}$ and σ^2_{MP} as

$$\begin{aligned} \alpha_i^{\mathrm{new}} &= \frac{1 - \alpha_i \boldsymbol{\Sigma}_{ii}}{\mu_i^2}, \\ \left(\sigma^2\right)^{\mathrm{new}} &= \frac{\|t - \boldsymbol{\varphi}\boldsymbol{\mu}\|^2}{N - \sum_i (1 - \alpha_i \boldsymbol{\Sigma}_{ii})}, \end{aligned} \tag{14}$$

where $\boldsymbol{\Sigma}_{ii}$ is the ith diagonal element of the posterior weight covariance.

Assume that $\mathbf{x}_*$ is a new input and the probability distribution of the output t_* is obtained by

$$\begin{aligned} &p\left(t_* \mid \mathbf{t}, \boldsymbol{\alpha}_{\mathrm{MP}}, \sigma^2_{\mathrm{MP}}\right) \\ &\quad = \int p\left(t_* \mid \mathbf{w}, \sigma^2_{\mathrm{MP}}\right) p\left(\mathbf{w} \mid \mathbf{t}, \boldsymbol{\alpha}_{\mathrm{MP}}, \sigma^2_{\mathrm{MP}}\right) d\mathbf{w}. \end{aligned} \tag{15}$$

It can be easily obtained for both integrated terms are Gaussian, and the result is also a Gaussian form

$$p\left(t_* \mid \mathbf{t}, \boldsymbol{\alpha}_{\mathrm{MP}}, \sigma^2_{\mathrm{MP}}\right) = N\left(t_* \mid y_*, \sigma_*^2\right). \tag{16}$$

The mean and the variance are

$$\begin{aligned} y_* &= \boldsymbol{\mu}^T \varphi(\mathbf{x}_*), \\ \sigma_*^2 &= \sigma^2_{\mathrm{MP}} + \varphi(\mathbf{x}_*)^T \boldsymbol{\Sigma} \varphi(\mathbf{x}_*). \end{aligned} \tag{17}$$

Gaussian radial basis function is selected as the kernel function for its powerful nonlinear processing capability, and the function is defined as

$$K(\mathbf{x}, \mathbf{x}_i) = \exp\left[-\frac{(\mathbf{x} - \mathbf{x}_i)^2}{2\gamma^2}\right], \tag{18}$$

where γ is the width factor which needs to be predetermined for it is crucially important to the predict performance.

3.2. DE Algorithm. DE algorithm is a population-based and stochastic search approach [18] and has shown superior performance on nonlinear, nonconvex, and nondifferentiable optimization problems [19–21]. DE algorithm starts with an initial population vector, which is randomly generated in a solution space. Assume that N is the population size and $X_{ri,G}$ $(i = 1, 2, \ldots, N)$ is a solution vector of the generation G.

For the classical DE algorithm, mutation and crossover are utilized to generate trial vectors, and selection is used to select the better vectors.

Mutation. For each vector $X_{ri,G}$, a mutant vector $V_{i,G}$ is generated by

$$V_{i,G} = X_{r1,G} + F\left(X_{r2,G} - X_{r3,G}\right) \quad r1 \neq r2 \neq r3 \neq i, \tag{19}$$

where $r1$, $r2$, and $r3$ are random integer indexes selected from $\{1, 2, \ldots, N\}$; F is the scale fact which determines the amplification of the difference vector $(X_{r2,G} - X_{r3,G})$, and $F \in [0, 2]$.

Crossover. The crossover operation refers to yielding the trial vector by using the mutant vector $V_{i,G}$ and target vector $X_{i,G}$:

$$U_{ji,G} = \begin{cases} V_{ji,G}, & \text{rand}_j \leq C_r \text{ or } j = k \\ X_{ji,G}, & \text{otherwise,} \end{cases} \tag{20}$$

where $j = 1, 2, \ldots, D$, and D is the problem dimension; $C_r \in [0, 1]$ is the predefined crossover constant; rand_j is the jth evaluation which is randomly generated between 0 and 1; $k \in \{1, 2, \ldots, D\}$ and it is a random index.

Selection. Assume that $f(x)$ is a minimization problem. The greedy selection scheme is defined as

$$X_{i,G+1} = \begin{cases} U_{i,G}, & \text{if } f\left(U_{i,G}\right) < f\left(X_{i,G}\right) \\ X_{i,G}, & \text{otherwise.} \end{cases} \tag{21}$$

The above three steps are repeated until reaching the terminal condition. Then the best vector with minimum fitness value is exported as the result.

3.3. Steps of Optimization. DE-RVM refers to the RVM with width factor optimized by DE algorithm. Mean square error (MSE) is used as the fitness function:

$$\text{MSE} = \frac{\sum_{\delta=1}^{H} \left[z^*(\delta) - z(\delta)\right]^2}{H}, \tag{22}$$

where MSE represents the deviate degree of the predicted data and the original data; $\delta = 1, 2, \ldots, H$, and H is the length of the original data; $z(\delta)$ and $z^*(\delta)$ are the original data and predicted data, respectively.

The optimization target is to minimize the MSE value, and the optimizing steps are described as follows:

(1) Initialize the DE algorithm parameters, which include the population size, scale factor, crossover rate, and maximum generation.

(2) Produce the mutant vector and trial vector according to (19) and (20).

(3) Determine the next generation vector according to (21).

(4) Repeat steps (2) and (3) until the terminated criterion is met.

(5) Output the optimized value to the RVM and exit the program.

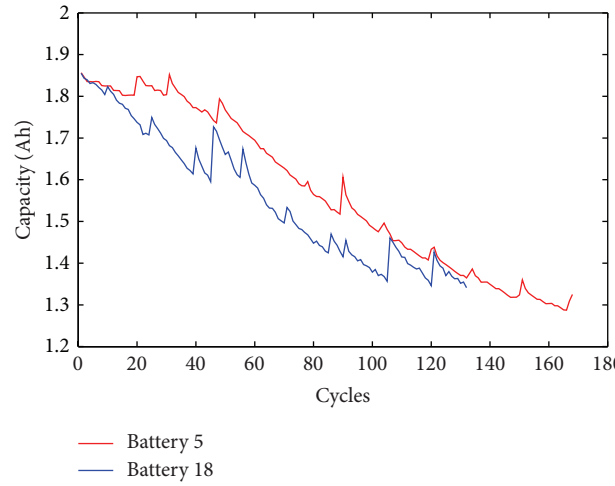


Figure 1: Measured capacity data of batteries.

4. Prognostics Experiment

4.1. Experiment Data. An experiment is conducted to demonstrate the proposed capacity prognostics approach, and the data were obtained from data repository of NASA Ames Prognostics Center of Excellence [22]. In the data collected procedure, lithium-ion batteries were working under three different operational profiles: charge, discharge, and impedance with a temperature of 25°C. Charging was performed at a 1.5 A constant current until the battery voltage reached 4.2 V and then maintaining the 4.2 V constant voltage until the current dropped to 20 mA. Discharging was running at a 2 A constant current until the battery voltage felled to 2.7 and 2.5 V, which were corresponding to batteries 5 and 18, respectively. Impedance measurement was implemented with an electrochemical impedance spectroscopy frequency sweep ranging from 0.1 Hz to 5 kHz. Repeated charge and discharge cycles led to the accelerated aging of batteries while impedance measurements discovered the changes of the internal battery parameters with aging progresses. The experiments were terminated when the capacity of batteries reached its end-of-life (EOL) threshold, which was about 70% rated capacity. In the experiments, each nominal capacity of lithium-ion battery is 2 Ah and the EOL threshold is set to 1.38 Ah. The lithium-ion batteries 5 and 18 capacity data are shown in Figure 1. It can be observed that the capacity generally degrades with usage for the reason of irreversible physical and chemical changes and at some cycle increases rapidly and shortly due to the impact of disturbances, measurement errors, stochastic load, or other unknown behaviors in the batteries. The length of batteries 5 and 18 capacity data are 166 cycles and 132 cycles, respectively. Meanwhile, their actual cycle lives are 129 and 114, respectively.

4.2. Experiment Procedure. The experiment includes a battery 5 capacity prognostics case and a battery 18 capacity

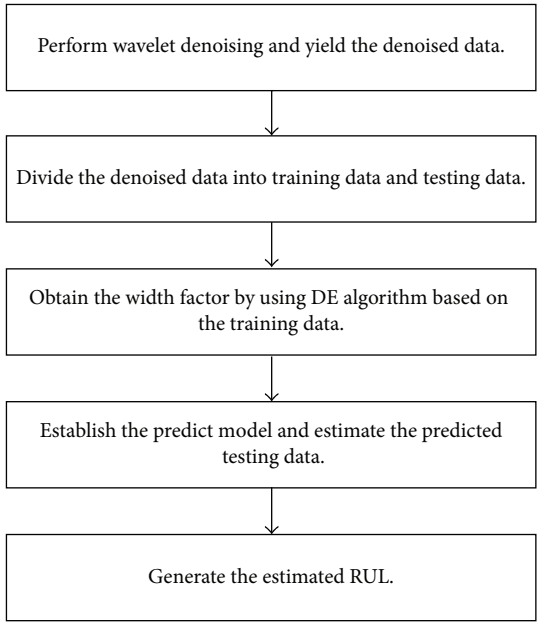


Figure 2: Flowchart of predict steps.

prognostics case. The detailed predict steps of each case are shown in Figure 2 and described as follows:

(1) Perform wavelet denoising based on the measured data and obtain the denoised data.

(2) Separate the denoised data into training data and testing data. The lengths of the training data in the two battery cases are set to 80 and 70, respectively. Therefore, the lengths of the testing data in two cases are 88 and 62, respectively.

(3) By using DE algorithm, a width factor is generated based on the training data.

(4) A predict model is constructed by RVM with adopting the optimized width factor, and the predicted testing data are estimated.

(5) Generate the estimated RUL.

4.3. Experiment Results and Analysis. Wavelet denoising implemented twice with different thresholds is employed to process the measured capacity data. Figure 3 displays the capacity data wavelet denoised with the sqtwolog threshold, and strong peak pulses are weakened obviously compared to Figure 1. Then the denoised capacity data are processed by using wavelet denoising with the minimax threshold to remove the weak noise. The denoised capacity data are shown in Figure 4 and the trajectory of the denoised data is continuous and smooth.

The DE algorithm population size and maximum generation are set to 30 and 100, respectively; F is equal to 0.6; C_r is linearly reduced from 0.9 to 0.3. Figure 5 shows the width factor optimization procedures by using DE algorithm based on batteries 5 and 18 training data, respectively. The corresponding optimized wider factors are 0.5009 and 0.2778 in the two battery cases, respectively.

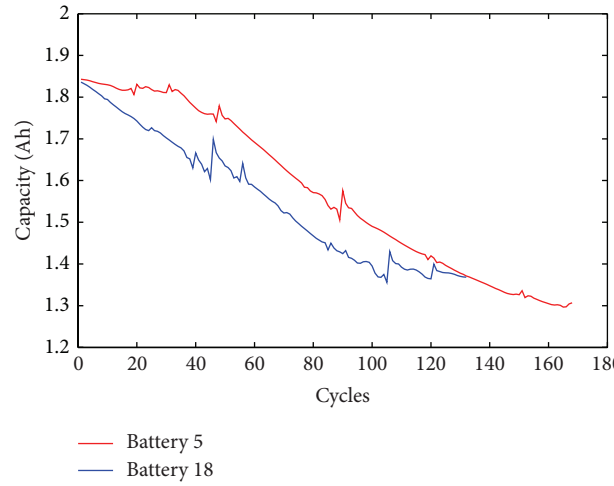


Figure 3: The batteries' capacity data wavelet denoised with the sqtwolog threshold.

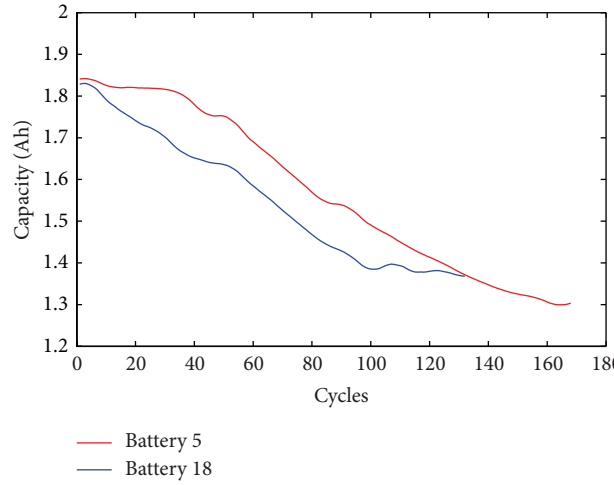


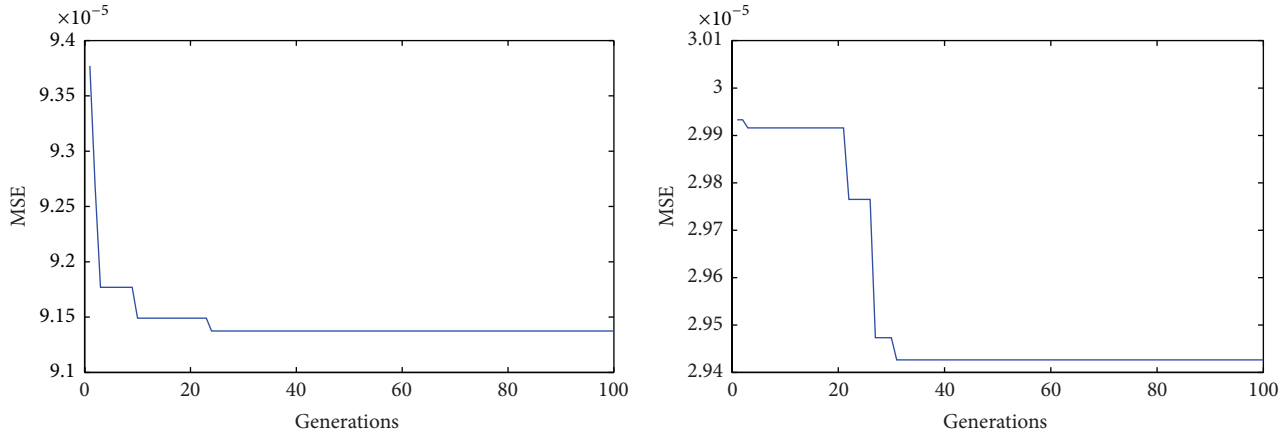
Figure 4: The batteries' capacity data wavelet denoised with the minimax threshold further.

Adopting the optimized width factor, RVM is used to perform battery capacity prognostics. In order to quantify the prognostic performance, absolute error (AE), and MSE between the original testing data and the predicted testing data, relative accuracy (RA) and $\alpha - \lambda$ accuracy [23] are employed as the measure metrics. The $\alpha - \lambda$ accuracy is applied to verify whether the estimated RUL is within the confidence interval defined by α. The metrics are defined as

$$\begin{aligned} \text{AE} &= \text{RUL}_{\text{es}} - \text{RUL}, \\ \text{RA} &= 1 - \frac{\left|\text{RUL}_{\text{es}} - \text{RUL}\right|}{\text{RUL}}, \\ \alpha - \lambda \text{ accuracy} &= \begin{cases} \text{Yes} & \text{if } \text{RUL}_{\text{es}} \in [C_1, C_2] \\ \text{No} & \text{if others,} \end{cases} \end{aligned} \tag{23}$$

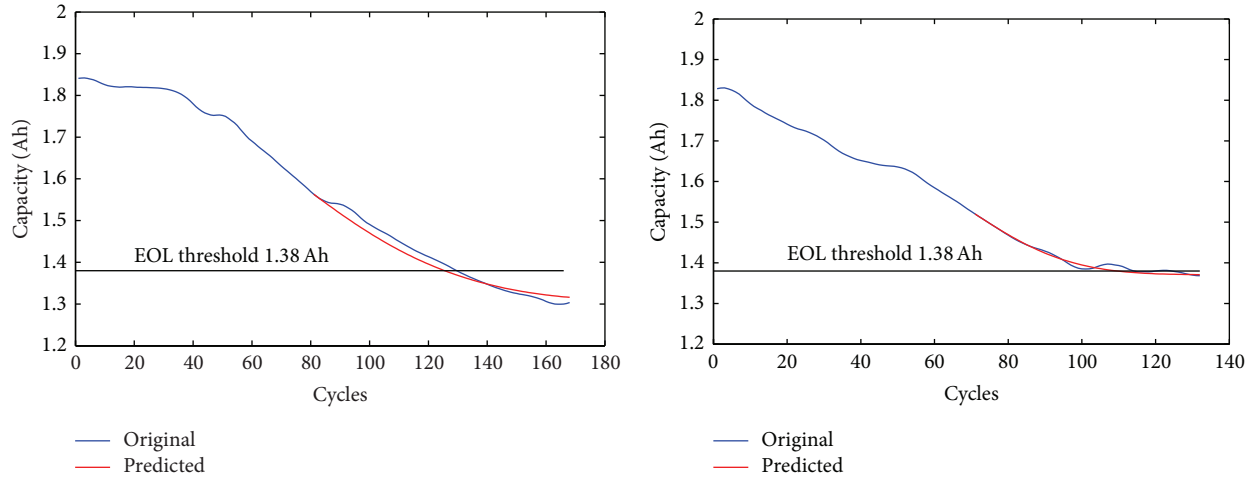
Table 1: Actual RULs, estimated RULs, AEs, RAs, and MSEs of two cases.

Case	Estimated RUL	Actual RUL	AE	RA	MSE	$\alpha - \lambda$ accuracy
Battery 5	45	49	−4	91.8%	$1.4178e-04$	Yes
Battery 18	40	44	−4	90.9%	$3.9249e-05$	Yes



(a) Battery 5 case (b) Battery 18 case

Figure 5: The width factor optimization procedures.



(a) Battery 5 case (b) Battery 18 case

Figure 6: Prediction results.

where RUL_{es} refers to the estimated RUL and RUL denotes the actual RUL; C_1 and C_2 are confidence intervals which equal RUL $*$ $(1 - \alpha)$ and RUL $*$ $(1 + \alpha)$, respectively; α is a bound which is set to 0.1 in the experiment.

Actual RULs in the two battery cases are 49 and 44, respectively. The predictions are displayed in Figure 6. Estimated RULs, actual RULs, AEs, RAs, MSEs, and $\alpha - \lambda$ accuracy of two cases are shown in Table 1. As can be seen from Figure 6, the DE-RVM predicts the trend of capacity degradation trajectories of the two cases successfully. Meanwhile this can also be verified by the MSEs in Table 1, which are pretty low in the two cases and this denotes that the predicted testing data are close to the original testing data. RAs are all beyond 90% in two cases which implies high prediction accuracies produced by the DE-RVM. Meanwhile, the estimated RULs in the two cases are both within the confidence interval as the last row shows.

For the purpose of validating the predict performance of the presented prognostics approach, the DE-RVM approach is compared with ANN optimized by DE algorithm (DE-ANN) [24] approach and SVM improved by DE algorithm (DE-SVM) [25] approach. The denoised data of batteries 5 and 18 are used as experiment data. The assessment index adopts RA and MSE. In order to avoid accidental accident in the experiment, each approach is run 10 times and mean results are shown in Table 2. As can be seen from the table,

Table 2: RAs and MSEs of the referenced approaches.

Case	DE-ANN		DE-SVM	
	RA	MSE	RA	MSE
Battery 5	85.7%	$2.7684e-04$	87.8%	$2.1507e-04$
Battery 18	81.8%	$7.3153e-05$	86.4%	$5.8236e-05$

the DE-RVM provides smaller MSE than the DE-ANN and DE-SVM which implies that the data predicted by the DE-RVM are more close to the original data. Meanwhile, the DE-RVM yields higher RA than the DE-ANN and the DE-SVM which characterizes that the DE-RVM can output more accurate prediction than the other two approaches. It can be concluded that the DE-RVM approach significantly outperforms the DE-ANN approach and the DE-SVM approach on the problem of the battery capacity prognostics.

5. Conclusions

The gradual decreased capacity of lithium-ion batteries has been used as the SOH indicator in the work. For the reason of the measured battery capacity data often suffering from the different levels of noise pollution, a wavelet denoising method with different thresholds has been presented to generate the denoised data.

The RVM with its width factor optimized by DE algorithm has been used for battery capacity prognostics. Two battery case results have validated that the approach can predict the trend of capacity degradation trajectory closely and estimate the battery RUL accurately. Meanwhile an extend experiment has demonstrated that the proposed DE-RVM approach has higher predict accuracy than the referenced approaches in the battery capacity prognostics.

Conflict of Interests

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

Acknowledgments

This work was supported by the National Natural Science Funds of China for Distinguished Young Scholar under Grant no. 50925727, the National Defense Advanced Research Project Grant nos. C1120110004 and 9140A27020211DZ5102, the Key Grant Project of Chinese Ministry of Education under Grant no. 313018, Anhui Provincial Science and Technology Foundation of China under Grant no. 1301022036, the Fundamental Research Funds for the Central Universities nos. 2012HGCX0003 and 2014HGCH0012, National Natural Science Foundation of China nos. 61401139, 51407054, and 61403115, and Anhui Provincial Natural Science Foundation no. 1508085QE85.

References

[1] S. Yuan, H. Wu, and C. Yin, "State of charge estimation using the extended Kalman filter for battery management systems based on the ARX battery model," *Energies*, vol. 6, no. 1, pp. 444–470, 2013.

[2] W. He, N. Williard, M. Osterman, and M. Pecht, "Prognostics of lithium-ion batteries using extended Kalman filtering," in *Proceedings of the IMAPS Advanced Technology Workshop on High Reliability Microelectronics for Military Applications*, pp. 17–19, Linthicum Heights, Md, USA, May 2011.

[3] S. Tang, C. Yu, X. Wang, X. Guo, and X. Si, "Remaining useful life prediction of lithium-ion batteries based on the wiener process with measurement error," *Energies*, vol. 7, no. 2, pp. 520–547, 2014.

[4] W. Xian, B. Long, M. Li, and H. Wang, "Prognostics of lithium-ion batteries based on the verhulst model, particle swarm optimization and particle filter," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 1, pp. 2–17, 2014.

[5] B. Saha, K. Goebel, S. Poll, and J. Christophersen, "Prognostics methods for battery health monitoring using a Bayesian framework," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 2, pp. 291–296, 2009.

[6] M. E. Orchard, P. Hevia-Koch, B. Zhang, and L. Tang, "Risk measures for particle-filtering-based state-of-charge prognosis in lithium-ion batteries," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 11, pp. 5260–5269, 2013.

[7] W. X. Shen, C. C. Chan, E. W. C. Lo, and K. T. Chau, "A new battery available capacity indicator for electric vehicles using neural network," *Energy Conversion and Management*, vol. 43, no. 6, pp. 817–826, 2002.

[8] A. Al-Alawi, S. M Al-Alawi, and S. M Islam, "Predictive control of an integrated PV-diesel water and power supply system using an artificial neural network," *Renewable Energy*, vol. 32, no. 8, pp. 1426–1439, 2007.

[9] A. Widodo, M.-C. Shim, W. Caesarendra, and B.-S. Yang, "Intelligent prognostics for battery health monitoring based on sample entropy," *Expert Systems with Applications*, vol. 38, no. 9, pp. 11763–11769, 2011.

[10] T. Hansen and C. J. Wang, "Support vector based battery state of charge estimator," *Journal of Power Sources*, vol. 141, no. 2, pp. 351–358, 2005.

[11] H. Li, D. Pan, and C. P. Chen, "Intelligent prognostics for battery health monitoring using the mean entropy and relevance Vector machine," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 44, pp. 851–862, 2014.

[12] D. Wang, Q. Miao, and M. Pecht, "Prognostics of lithium-ion batteries based on relevance vectors and a conditional three-parameter capacity degradation model," *Journal of Power Sources*, vol. 239, pp. 253–264, 2013.

[13] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[14] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *The Journal of Machine Learning Research*, vol. 1, no. 3, pp. 211–244, 2001.

[15] W. Caesarendra, A. Widodo, and B. S. Yang, "Application of relevance vector machine and logistic regression for machine degradation assessment," *Mechanical Systems and Signal Processing*, vol. 24, no. 4, pp. 1161–1171, 2010.

[16] Z. Chen, "An overview of bayesian methods for neural spike train analysis," *Computational Intelligence and Neuroscience*, vol. 2013, Article ID 251905, 17 pages, 2013.

[17] C. Zhang, Y. He, L. Yuan, and F. Deng, "A novel approach for analog circuit fault prognostics based on improved RVM," *Journal of Electronic Testing*, vol. 30, no. 3, pp. 343–356, 2014.

[18] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[19] P. Rocca, G. Oliveri, and A. Massa, "Differential evolution as applied to electromagnetics," *IEEE Antennas and Propagation Magazine*, vol. 53, no. 1, pp. 38–49, 2011.

[20] A. A. Abou El Ela, M. A. Abido, and S. R. Spea, "Differential evolution algorithm for optimal reactive power dispatch," *Electric Power Systems Research*, vol. 81, no. 2, pp. 458–464, 2011.

[21] A. Bhattacharya and P. K. Chattopadhyay, "Hybrid differential evolution with biogeography-based optimization for solution of economic load dispatch," *IEEE Transactions on Power Systems*, vol. 25, no. 4, pp. 1955–1964, 2010.

[22] B. Saha and K. Goebel, "Battery Data Set," NASA Ames Prognostics Data Repository, NASA Ames, Moffett Field, Calif, USA, 2007, http://ti.arc.nasa.gov/tech/dash/pcoe/prognostic-data-repository/.

[23] A. Saxena, J. Celaya, B. Saha, S. Saha, and K. Goebel, "Metrics for offline evaluation of prognostic performance," *International Journal of Prognostics and Health Management*, vol. 1, no. 1, 20 pages, 2010.

[24] S. H. Yang, U. Natarajan, M. Sekar, and S. Palani, "Prediction of surface roughness in turning operations by computer vision using neural network trained by differential evolution algorithm," *The International Journal of Advanced Manufacturing Technology*, vol. 51, no. 9–12, pp. 965–971, 2010.

[25] J. Wang, L. Li, D. Niu, and Z. Tan, "An annual load forecasting model based on support vector regression with differential evolution algorithm," *Applied Energy*, vol. 94, pp. 65–70, 2012.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 583759, 11 pages
http://dx.doi.org/10.1155/2015/583759

Research Article

An Enhanced Differential Evolution with Elite Chaotic Local Search

Zhaolu Guo,[1] Haixia Huang,[2] Changshou Deng,[3] Xuezhi Yue,[1] and Zhijian Wu[4]

[1] *Institute of Medical Informatics and Engineering, School of Science, Jiangxi University of Science and Technology, Ganzhou 341000, China*
[2] *School of Literature and Law, Jiangxi University of Science and Technology, Ganzhou 341000, China*
[3] *School of Information Science and Technology, Jiujiang University, Jiujiang 332005, China*
[4] *State Key Laboratory of Software Engineering, Wuhan University, Wuhan 430072, China*

Correspondence should be addressed to Zhaolu Guo; gzl@whu.edu.cn

Received 8 October 2014; Accepted 27 April 2015

Academic Editor: Rafik Aliyev

Differential evolution (DE) is a simple yet efficient evolutionary algorithm for real-world engineering problems. However, its search ability should be further enhanced to obtain better solutions when DE is applied to solve complex optimization problems. This paper presents an enhanced differential evolution with elite chaotic local search (DEECL). In DEECL, it utilizes a chaotic search strategy based on the heuristic information from the elite individuals to promote the exploitation power. Moreover, DEECL employs a simple and effective parameter adaptation mechanism to enhance the robustness. Experiments are conducted on a set of classical test functions. The experimental results show that DEECL is very competitive on the majority of the test functions.

1. Introduction

Numerous problems in science and engineering can be converted into optimization problems. Therefore, it is of significance both in theory and in engineering applications to develop effective and efficient optimization algorithms for solving complex problems of science and engineering. Differential evolution (DE), proposed by Storn and Price in 1997 [1], is a simple yet effective global optimization algorithm. According to frequently reported theoretical and experimental studies, DE has exhibited competitive performance than many other evolutionary algorithms in terms of both convergence speed and solution precision over several benchmark functions and real-life problems [2–4]. Due to its simplicity, easy implementation, and efficiency, DE has stimulated many researchers' interests since its development. Therefore, it has become a hot research topic in evolutionary computation over the past decades [5–7].

However, its search ability should be further enhanced to obtain better solutions when DE is used to solve various real-life optimization problems [2, 8, 9]. Particularly, DE may suffer from premature convergence and/or slow convergence when solving complex multimodal optimization problems. In order to improve the performance of the conventional DE, a number of DE variants have been proposed in recent decades [2, 6, 10]. Recognizing that the performance of DE depends on the control parameters, Brest et al. [11] presented a self-adaptive DE (jDE), in which both F and CR are created independently for each individual by an adaptive mechanism. Specifically, the new F is created by a random value from 0.1 to 0.9 with a probability 0.1 during the search process. Meanwhile, the new CR obtains a random value from 0.0 to 1.0 with a probability 0.1. Unlike jDE, JADE, proposed by Zhang and Sanderson [12], utilizes a distinct parameter adaptation mechanism, in which the new F and CR are created for each individual by a normal distribution and a Cauchy distribution, respectively. In addition, JADE learns knowledge from the recent successful F and CR and applies the learned knowledge for creating new F and CR. Identifying that both the mutation strategies and their associated control parameters can directly influence the performance of DE, Qin et al. [7] proposed a novel self-adaptive DE, SaDE, which adaptively tunes the trial vector generation strategies and their associated control parameter values by extracting

knowledge from the previous search process in generating promising solutions. Mallipeddi et al. [13] introduced an improved DE with ensemble of parameters and mutation strategies (EPSDE), which employs a pool of diverse trial vector generation strategies and a pool of values for the control parameters F and CR. By incorporating an opposition-based learning strategy into the traditional DE for population initialization and generating new solutions, Rahnamayan et al. [14] proposed an opposition-based DE (ODE). The experimental results confirmed that the opposition-based learning strategy can improve the convergence speed and the solution accuracy of DE. Further, Wang et al. [15] improved the opposition-based learning strategy, proposed a generalized opposition-based learning strategy, and presented an enhanced DE with generalized opposition-based learning strategy (GODE). Jia et al. [16] presented an effective memetic DE algorithm, DECLS, which utilizes a chaotic local search with a shrinking strategy to improve the search ability. Experimental results indicated that the performance of the canonical DE is significantly improved by the chaotic local search. Recently, Wang et al. [17] proposed a composite DE, called CoDE, the main idea of which is to randomly combine several well studied trial vector generation strategies with a number of control parameter settings highly recommended by other researchers at each generation to create new trial vectors. Experimental results on all the CEC2005 contest test instances show that CoDE is very competitive.

Although there already exist many DE variants for solving complex optimization problems, according to the no free lunch (NFL) theory [18], the performance of DE for some benchmark functions and real-life problems should be further enhanced to obtain better solutions. Moreover, many studies have revealed that embedding local search strategy can greatly enhance the search ability of DE [14, 16, 19]. Motivated by these considerations, in order to promote the performance of DE on complex optimization problems, this study proposes an enhanced differential evolution with elite chaotic local search, called DEECL. In DEECL, we utilize a chaotic search strategy based on the heuristic information from the elite individuals to promote the exploitation power. Further, we also design a simple and effective parameter adaptation mechanism to enhance the robustness.

The rest of the paper is organized as follows. The conventional DE is introduced in Section 2. Section 3 presents the enhanced DE. Numerical experiments are presented in Section 4 for the comparison and analysis. Finally, the paper is concluded in Section 5.

2. Differential Evolution

Without loss of generality, only minimization problems are considered in this study. We suppose that the objective function to be minimized is Min $f(X)$, $X = [X_1, X_2, \ldots, X_D]$, and the search space is

$$\Omega = \prod_{j=1}^{D} \left[\mathrm{LB}_j, \mathrm{UB}_j\right], \tag{1}$$

where D is the number of dimensions of the problem, LB_j and UB_j denote the lower and upper boundaries of the search space, respectively.

Similar to other evolutionary algorithms, DE also has a simple structure, only including three simple operators, namely, mutation, crossover, and selection operators [2]. In the initial phase, DE creates an initial population $P(t) = \{X_i^t\}$, which is randomly generated from the search space, where $X_i^t = [X_{i,1}^t, X_{i,2}^t, \ldots, X_{i,D}^t]$, $i = 1, 2, \ldots, NP$; NP is the population size and t is the generation. After initialization, the mutation and crossover operators are performed to create the trial vectors, and then the selection operator is utilized to select the better one between the offspring individual and the parent individual for the next generation. DE performs these steps repeatedly to converge toward the global optima until the terminating criterion is reached [20]. In the following subsections, the evolutionary operators of DE will be introduced in detail.

2.1. Mutation Operator. In the mutation operator, a mutant vector V_i^t is created by using a predetermined mutation strategy for each individual X_i^t, namely, target vector, in the current population [17]. DE has many mutation strategies used in its implementations, such as DE/rand/1, DE/best/1, DE/rand-to-best/1, DE/best/2 and DE/rand/2 [2]. Among these mutation strategies, DE/rand/1 is the most frequently used mutation strategy, which is expressed as follows [1]:

$$V_i^t = X_{r1}^t + F \times \left(X_{r2}^t - X_{r3}^t\right), \tag{2}$$

where $r1$, $r2$, and $r3$ are randomly selected from the set $\{1, 2, \ldots, NP\} \setminus \{i\}$, and they are mutually different from each other. F is called as scaling factor, amplifying the difference vector $X_{r2}^t - X_{r3}^t$.

2.2. Crossover Operator. Following mutation, a trial vector U_i^t is generated by executing the crossover operator for each pair of target vector X_i^t and its corresponding mutant vector V_i^t [2]. Binomial crossover is the most commonly used crossover operators in current popular DE. The binomial crossover is described as follows [1]:

$$U_{i,j}^t = \begin{cases} V_{i,j}^t, & \text{if rand}(0,1) < \mathrm{CR} \text{ or } j == j\mathrm{rand} \\ X_{i,j}^t, & \text{otherwise,} \end{cases} \tag{3}$$

where rand$(0, 1)$ is generated for each j and takes a value from 0.0 to 1.0 in a uniformly random manner, and $\mathrm{CR} \in [0, 1]$ is the crossover probability, which limits the number of parameters inherited from the mutant vector V_i^t. The integer $j\mathrm{rand}$ is randomly chosen from the range $[1, D]$, which guarantees that at least one parameter of the trial vector U_i^t is inherited from the mutant vector V_i^t [7].

2.3. Selection Operator. Like the genetic algorithm, the selection process of DE is also based on the Darwinian law of survival of the fittest. The selection process is performed in order to choose the more excellent individuals for the

```
t = 0;
FES = 0;
/* Initialize the population */
for i = 1 to NP do
  for j = 1 to D do
    X^t_{i,j} = LB_j + rand(0, 1) × (UB_j − LB_j);
  end for
  Evaluate individual X^t_i;
  FES = FES + 1;
end for
while FES < MAX_FES do
  for i = 1 to NP do
    Choose three mutually different integers r1, r2, r3 from
    the set {1, 2, ..., NP} \ {i} in a random manner;
    jrand = randint(1, D);
    for j = 1 to D do
      if rand(0, 1) < CR or j == jrand then
        U^t_{i,j} = X^t_{r1,j} + F × (X^t_{r2,j} − X^t_{r3,j});
      else
        U^t_{i,j} = X^t_{i,j};
      end if
    end for
    /* Selection step */
    if f(U^t_i) ≤ f(X^t_i) then
      X^{t+1}_i = U^t_i;
      if f(U^t_i) < f(X^t_Best) then
        X^t_Best = U^t_i;
      end if
    else
      X^{t+1}_i = X^t_i;
    end if
    FES = FES + 1;
  end for
  t = t + 1;
end while
```

Algorithm 1: DE algorithm.

next generation. For minimization problems, the selection operator can be defined in the following form [1]:

$$X_i^{t+1} = \begin{cases} U_i^t, & \text{if } f\left(U_i^t\right) \le f\left(X_i^t\right) \\ X_i^t, & \text{otherwise,} \end{cases} \tag{4}$$

where $f(X_i^t)$ and $f(U_i^t)$ indicate the fitness values of the target vector X_i^t and its corresponding trial vector U_i^t, respectively.

2.4. Algorithmic Framework of DE. Based on the above elaborate introduction of the DE's operators, we present the framework of DE with DE/rand/1/bin strategy in Algorithm 1, where FES is the number of fitness evaluations, Max_FES is the maximum number of evaluations, rand(0,1) indicates a random real number in the range $[0, 1]$, randint$(1, D)$ represents a random integer in the range $[1, D]$, and X_{Best}^t is the global best individual found so far.

3. Proposed Approach

3.1. Motivations. DE has been demonstrated to yield superior performance for solving various real-world optimization problems [21–23]. However, it tends to suffer from premature convergence and/or slow convergence when solving complex optimization problems [6, 24]. To enhance the performance of DE, many researchers have proposed various improved DE algorithms during the past decade [25–27]. Among the DE variations, memetic method is a promising approach to improve the performance of the traditional DE, which utilizes various local search strategies, such as chaotic search strategy [16], simplex crossover search strategy [19], and orthogonal search strategy [28], to strengthen the exploitation ability of the traditional DE and consequently accelerate the convergence speed. Among the local search strategies commonly used in memetic DE, chaotic search strategy is inspired by the chaos phenomenon in nature. Chaos is a classic nonlinear dynamical system, which is widely known as a system with the properties of ergodicity, randomicity, and sensitivity to its initial conditions [16, 29, 30]. Due to its ergodicity and randomicity, a chaotic system can randomly generate a long-time sequence which is able to traverse through every state of the system and every state is generated only once if given a long enough time period [16, 31]. Taking advantage of the well-known characteristics of the chaotic systems, researchers have proposed many chaotic search strategies for optimizing various problems [16, 32–34]. However, to the best of our knowledge, among many chaotic search strategies, they pay more attention to the characteristics of the ergodicity and randomicity of the chaotic system. Therefore, the exploration capacity can be indeed improved. However, in order to maintain a balance between exploration and exploitation, the exploitation ability of the chaotic search strategy should be further enhanced. Thus, when designing a relatively comprehensive chaotic search strategy, we should further integrate more heuristic information into the chaotic search strategy to promote its exploitation power. Generally, the elite individuals in the current population known as a promising search direction toward the optimum are the favorable source that can be employed to enhance the exploitation ability. Based on these considerations, we present an elite chaotic search strategy, which not only utilizes the characteristics of the ergodicity and randomicity of the chaotic system, but also merges the superior information of the current population into the chaotic search process.

3.2. Elite Chaotic Search. In many chaotic search strategies, the Logistic chaotic function is utilized to generate a chaotic sequence, which is formulated as follows [16]:

$$\begin{aligned} &K^0 = \text{rand}\,(0, 1), \quad K^0 \neq 0.25, 0.5, 0.75, \\ &K^n = 4.0 \cdot K^{n-1} \cdot \left(1 - K^{n-1}\right), \quad n = 1, 2, \ldots, N, \end{aligned} \tag{5}$$

where K^0 is the initial value of the chaotic system, which is randomly generated from the range $[0, 1]$, but cannot be equal to 0.25, 0.5, or 0.75. K^n is the nth state of the chaotic system. As known, the initial state K^0 of the chaotic system

$n = 0$;
$N = D/5$;
Randomly choose an individual X_I^t from the current population;
$K^0 = \text{rand}(0, 1), K^0 \neq 0.25, 0.5, 0.75$;
$p = \text{rand}(2.0/NP, 0.1)$;
while $n < N$ do
  Randomly choose an individual $X_{p\text{Best}}^t$ from the top $100p\%$ individuals in the current population;
  for $j = 1$ to D do
    $E_{I,j}^n = X_{I,j}^t + K^n \times (X_{p\text{Best},j}^t - X_{I,j}^t)$;
    if $E_{I,j}^n > \text{UB}_j$ or $E_{I,j}^n < \text{LB}_j$ then
      $E_{I,j}^n = \text{rand}(\text{LB}_j, \text{UB}_j)$;
    end if
  end for
  if $f(E_I^n) < f(X_I^t)$ then
    $X_I^t = E_I^n$;
    break;
  end if
  $n = n + 1$;
  $K^n = 4.0 \cdot K^{n-1} \cdot (1 - K^{n-1})$;
  FES = FES + 1;
end while

Algorithm 2: Elite chaotic search operator.

is randomly produced. Due to its ergodicity and sensitivity to the initial state K^0, K^n is a random long-time sequence, which can traverse through every state of the system and every state is generated only once if N is large enough.

In order to enhance the exploitation ability of the traditional chaotic search strategy, we integrate the heuristic information learned from the elite individuals into the chaotic search strategy to promote the exploitation power. The proposed elite chaotic search strategy is defined by

$$E_I^n = X_I^t + K^n \times \left(X_{p\text{Best}}^t - X_I^t\right), \tag{6}$$

where X_I^t is an individual to be performed the elite chaotic search, which is randomly chosen from the current population. K^n is the chaotic sequence, where $n = 1, 2, \ldots, N$, $N = D/5$, and $X_{p\text{Best}}^t$ is an elite individual, which is randomly chosen from the top $100p\%$ individuals in the current population with $p = \text{rand}(2.0/NP, 0.1)$.

In the proposed elite chaotic search operator, an individual X_I^t is randomly selected from the current population to undergo the elite chaotic search strategy. After that, the initial value of the chaotic system takes a value from range $[0, 1.0]$ in a uniformly random manner. Then, an elite chaotic search procedure for individual X_I^t is repeatedly performed until finding a better solution than individual X_I^t or the number of iterations n is equal to N. The framework of the elite chaotic search operator is described in Algorithm 2.

3.3. Parameter Adaptation. Since the setting of control parameters can significantly influence the performance of DE, parameter adaptation mechanism is essential for an efficient DE [7, 11, 12]. To this end, we design a simple and effective parameter adaptation mechanism inspired by [11] into DEECL. In DEECL, each individual is independently associated with its own mutation factor F_i^t and crossover probability CR_i^t. For individual i, its control parameters F_i^t and CR_i^t are initialized to 0.5 and 0.9, respectively. Generally, a normal distribution with mean value 0.5 and standard deviation 0.3 is a promising adaptive approach for the mutation factor of DE [7], whereas Cauchy distribution is more favorable to diversify the mutation factors and thus avoid premature convergence [12]. Based on these considerations, at each generation, the new mutation factor NF_i^t associated with individual i is generated by a Cauchy distribution random real number with location parameter 0.5 and scale parameter 0.3 with probability 0.1. Additionally, following the suggestions in [11], the new crossover probability NCR_i^t associated with individual i acquires a random value from 0.0 to 1.0 with probability 0.1. Mathematically, the new control parameters NF_i^t and NCR_i^t associated with individual i for generating its corresponding trial vector U_i^t are obtained by

$$\begin{aligned} \text{NF}_i^t &= \begin{cases} \text{randc}\,(0.5, 0.3)\,, & \text{if rand}\,(0, 1) < 0.1 \\ F_i^t, & \text{otherwise,} \end{cases} \\ \text{NCR}_i^t &= \begin{cases} \text{rand}\,(0, 1)\,, & \text{if rand}\,(0, 1) < 0.1 \\ \text{CR}_i^t, & \text{otherwise,} \end{cases} \end{aligned} \tag{7}$$

where $\text{randc}(0.5, 0.3)$ is a Cauchy distribution random real number with location parameter 0.5 and scale parameter 0.3 and $\text{rand}(0, 1)$ is a uniformly random number within the range $[0, 1]$. After obtaining the new control parameters NF_i^t and NCR_i^t, the corresponding trial vector U_i^t are created

Table 1: The 13 classical test functions.

Function	Name	Initial range	$f_{\min}$
$f1$	Sphere Problem	$[-100, 100]^D$	0
$f2$	Schwefel's Problem 2.22	$[-10, 10]^D$	0
$f3$	Schwefel's Problem 1.2	$[-100, 100]^D$	0
$f4$	Schwefel's Problem 2.21	$[-100, 100]^D$	0
$f5$	Rosenbrock's Function	$[-30, 30]^D$	0
$f6$	Step Function	$[-100, 100]^D$	0
$f7$	Quartic Function with Noise	$[-1.28, 1.28]^D$	0
$f8$	Schwefel's Problem 2.26	$[-500, 500]^D$	0
$f9$	Rastrigin's Function	$[-5.12, 5.12]^D$	0
$f10$	Ackley's Function	$[-32, 32]^D$	0
$f11$	Griewank Function	$[-600, 600]^D$	0
$f12$	Penalized Function 1	$[-50, 50]^D$	0
$f13$	Penalized Function 2	$[-50, 50]^D$	0

by using the new control parameters NF_i^t and NCR_i^t. It is widely acknowledged that better control parameter values tend to produce better individuals that have a greater chance to survive and thus these values should be propagated to the next generations [12]. Therefore, in the selection step, the control parameters F_i^{t+1} and CR_i^{t+1} associated with individual i for the next generation are updated by

$$F_i^{t+1} = \begin{cases} \mathrm{NF}_i^t, & \text{if } f\left(U_i^t\right) < f\left(X_i^t\right) \\ F_i^t, & \text{otherwise,} \end{cases}$$
$$\mathrm{CR}_i^{t+1} = \begin{cases} \mathrm{NCR}_i^t, & \text{if } f\left(U_i^t\right) < f\left(X_i^t\right) \\ \mathrm{CR}_i^t, & \text{otherwise.} \end{cases} \quad (8)$$

From the above designed parameter adaptation mechanism, we can infer that the better control parameters of DEECL can be propagated to the next generations. Therefore, the control parameters of DEECL can be adaptively tuned according to the feedback from the search process.

4. Numerical Experiments

4.1. Experimental Setup. In order to assess the performance of the proposed DEECL, we use 13 classical test functions ($f1$–$f13$) that are widely used in the evolutionary computation community [8, 12, 35] to verify the effectiveness of the proposed DEECL. We describe these test functions in Table 1. Among these test functions [35], $f1$–$f4$ are continuous unimodal functions. $f5$ is the Rosenbrock function which is unimodal for $D = 2$ and 3; however, it may have multiple minima in high dimension cases [36]. $f6$ is a discontinuous step function, and $f7$ is a noisy function. $f8$–$f13$ are multimodal functions and they exist many local minima [35].

In all experiments, we set the number of dimensions D to 30 for all these test functions. We carry out 30 independent runs for each algorithm and each test function with 150,000 function evaluations (FES) as the termination criterion. Moreover, we record the average and standard deviation of the function error value ($f(x) - f(x^*)$) for estimating the performance of the algorithms, as recommended by [17], where x is the best solution gained by the algorithm in a run and x^* is the global optimum of the test function.

4.2. Benefit of the Two Components. There are two important components in the proposed DEECL: the proposed elite chaotic search strategy and the designed parameter adaptation mechanism. Accordingly, it is interesting to recognize the benefit of the two components of the proposed DEECL. For this purpose, we conduct experiments to compare the proposed DEECL with the traditional DE with DE/rand/1 strategy and two variants of DEECL, namely, DE with the proposed elite chaotic search strategy (DEwEC) and DE with the designed parameter adaptation mechanism (DEwPA). In the experiments, we set the population size of all the algorithms to 100. For the other parameters of DE and DEwEC, we set $F = 0.5$ and $\mathrm{CR} = 0.9$, following the suggestions in [11].

We present the experimental results of the above mentioned algorithms in Table 2. The best results among the four algorithms are highlighted in *boldface*. "Mean Error" and "Std Dev" indicate the mean and standard deviation of the function error values achieved in 30 independent runs, respectively. From the results of comparison between DE and DEwEC, DEwEC performs better than DE on all test functions with the exception of $f6$. On test function $f6$, both DE and DEwEC exhibit similar performance. In total, DEwEC is better than DE on twelve test functions. The results of comparison between DE and DEwEC indicate that our introduced elite chaotic search strategy is effective to enhance the performance of the traditional DE.

From the comparison of DE with DEwPA, DEwPA surpasses DE on all test functions except for $f5$ and $f6$. On test function $f6$, both DE and DEwPA demonstrate similar performance, whereas DE is better than DEwPA on test function $f5$. In summary, DEwPA outperforms DE on eleven test functions. The comparison of DE with DEwPA reveals that our designed parameter adaptation mechanism is capable of improving the efficiency of the traditional DE.

By incorporation of both the proposed elite chaotic search strategy and the designed parameter adaptation mechanism, DEECL achieves promising performance, which is better than other three DE algorithms on the majority of the test functions. To be specific, DEECL is better than DE, DEwEC, and DEwPA on eleven, nine, and ten test functions, respectively. DE, DEwEC, and DEwPA can outperform DEECL only on one test function. Comparison results suggest that both the introduced elite chaotic search strategy and the designed parameter adaptation mechanism demonstrate positive effect on the performance of DEECL. In addition, the comparison results confirm that the introduced elite chaotic search strategy and the designed parameter adaptation mechanism can help DE with both outperform DE with either or neither one on the majority of the test functions. Moreover, the introduced elite chaotic search strategy and the designed parameter adaptation mechanism work together to improve the performance of the traditional DE rather than contradict each other. The evolution of the average function error values derived from DE, DEwEC, DEwPA, and DEECL versus

Table 2: Experimental results of DE, DEwEC, DEwPA, and DEECL over 30 independent runs for the 13 test functions.

Function	DE	DEwEC	DEwPA	DEECL
	Mean ± Std Dev	Mean ± Std Dev	Mean ± Std Dev	Mean ± Std Dev
$f1$	$2.23E-16 \pm 2.50E-16$	$4.38E-24 \pm 3.95E-24$	$4.14E-33 \pm 4.76E-33$	**6.89E − 38 ± 6.06E − 38**
$f2$	$2.86E-08 \pm 1.26E-08$	$1.63E-12 \pm 6.75E-13$	$3.58E-20 \pm 2.80E-20$	**1.74E − 22 ± 1.21E − 22**
$f3$	$1.88E-01 \pm 6.12E-02$	$9.12E-02 \pm 1.03E-02$	$5.25E-02 \pm 5.94E-02$	**2.42E − 02 ± 3.44E − 02**
$f4$	$1.70E-01 \pm 2.13E-01$	$1.51E-02 \pm 2.10E-02$	$3.23E-04 \pm 1.55E-04$	**4.06E − 05 ± 3.05E − 05**
$f5$	$1.39E+01 \pm 8.74E-01$	**1.27E + 01 ± 5.12E + 01**	$2.46E+01 \pm 1.58E+01$	$2.95E+01 \pm 2.21E+01$
$f6$	**0.00E + 00 ± 0.00E + 00**	**0.00E + 00 ± 0.00E + 00**	**0.00E + 00 ± 0.00E + 00**	**0.00E + 00 ± 0.00E + 00**
$f7$	$8.82E-03 \pm 2.61E-03$	$2.15E-03 \pm 8.72E-04$	$6.52E-03 \pm 1.59E-03$	**1.17E − 03 ± 6.52E − 04**
$f8$	$7.31E+03 \pm 3.75E+02$	$5.04E+03 \pm 3.25E+02$	$1.53E-02 \pm 1.82E-12$	**1.34E − 02 ± 1.19E − 12**
$f9$	$1.77E+02 \pm 1.10E+01$	**0.00E + 00 ± 0.00E + 00**	**0.00E + 00 ± 0.00E + 00**	**0.00E + 00 ± 0.00E + 00**
$f10$	$5.93E-09 \pm 3.10E-09$	$5.22E-13 \pm 1.76E-13$	$4.35E-15 \pm 1.07E-15$	**4.00E − 15 ± 0.00E + 00**
$f11$	$6.33E-16 \pm 1.16E-15$	**0.00E + 00 ± 0.00E + 00**	**0.00E + 00 ± 0.00E + 00**	**0.00E + 00 ± 0.00E + 00**
$f12$	$2.20E-17 \pm 1.81E-17$	$2.21E-25 \pm 1.38E-25$	$1.61E-30 \pm 7.69E-48$	**1.57E − 32 ± 2.74E − 48**
$f13$	$8.26E-17 \pm 3.59E-17$	$4.23E-24 \pm 6.14E-24$	$1.46E-32 \pm 2.02E-33$	**1.36E − 32 ± 3.70E − 34**

the number of FES is plotted in Figure 1 for some typical test functions. As can be seen from Figure 1, DEECL converges faster than DE, DEwEC, and DEwPA.

4.3. Comparison with Other DE Variants. In order to verify the effectiveness of the proposed DEECL algorithm, we compare DEECL with the traditional DE and three other DE variants, namely, jDE [11], ODE [14], and DECLS [16]. In addition, jDE is a self-adaptive DE, in which both parameters F and CR are generated independently for each individual by an adaptive mechanism [11]. ODE is proposed by Rahnamayan et al. [14], which incorporates the opposition-based learning strategy into the traditional DE for population initialization and creating new solutions. DECLS is an effective memetic DE algorithm [16], which utilizes the chaotic local search strategy and an adaptive parameter control approaches similar to jDE [11] to improve the search ability. In the experiments, in order to have a fair comparison, we set the population size of all the algorithms to 100. The other parameter settings of these three DE variants are the same as in their original papers.

The mean and standard deviation of the function error values achieved by each algorithm for the 13 classical test functions are presented in Table 3. For convenience of analysis, the best results among the four DE algorithms are highlighted in *boldface*. In order to gain statistically significant conclusions, we conduct two-tailed t-tests at the significance level of 0.05 [28, 35] on the experimental results. The summary comparison results are described in the last three rows of Table 3. "+," "−," and "≈" suggest that DEECL is better than, worse than, and similar to the corresponding algorithm in terms of the two-tailed t-tests at the significance level of 0.05, respectively.

From Table 3, we can infer that DEECL achieves the better results than all the other four algorithms on the majority of the 13 classical test functions. Specifically, DEECL is significantly better than DE, jDE, ODE, and DECLS on eleven, seven, nine, and six test functions according to the two-tailed t-test, respectively. In addition, DEECL is similar to DE, jDE, ODE, and DECLS on one, five, two, and five test functions, respectively. DE and jDE surpasses DEECL only on one test function. Additionally, ODE and DECLS perform better than DEECL only on two test functions.

Overall, DEECL performs better than the traditional DE, jDE, ODE, and DECLS on the majority of the test functions. This can be because the proposed elite chaotic search strategy learning the heuristic information from the elite individuals can promote the exploitation power, and the designed parameter adaptation mechanism can enhance the robustness. The evolution of the average function error values derived from DE, jDE, ODE, DECLS, and DEECL versus the number of FES is plotted in Figure 2 for some typical test functions. It can be known from Figure 2 that DEECL converges faster than DE, jDE, ODE, and DECLS.

In order to compare the total performance of the five DE algorithms on the all 13 classical test functions, we carry out the average ranking of Friedman test on the experimental results following the suggestions in [37–39]. Table 4 presents the average ranking of the five DE algorithms on the all 13 classical test functions. We can sort these five DE algorithms by the average ranking into the following order: DEECL, DECLS, jDE, ODE, and DE. Therefore, DEECL obtains the best average ranking, and its total performance is better than that of the other four algorithms on the all 13 test instances.

5. Conclusions

DE is a popular evolutionary algorithm for the continuous global optimization problems, which has a simple structure yet exhibits efficient performance on various real-world engineering problems. However, according to the no free lunch (NFL) theory, the performance of DE should be further enhanced to obtain better solutions in some cases. In this paper, we propose an enhanced differential evolution with elite chaotic local search, called DEECL, which uses a chaotic search strategy based on the heuristic information

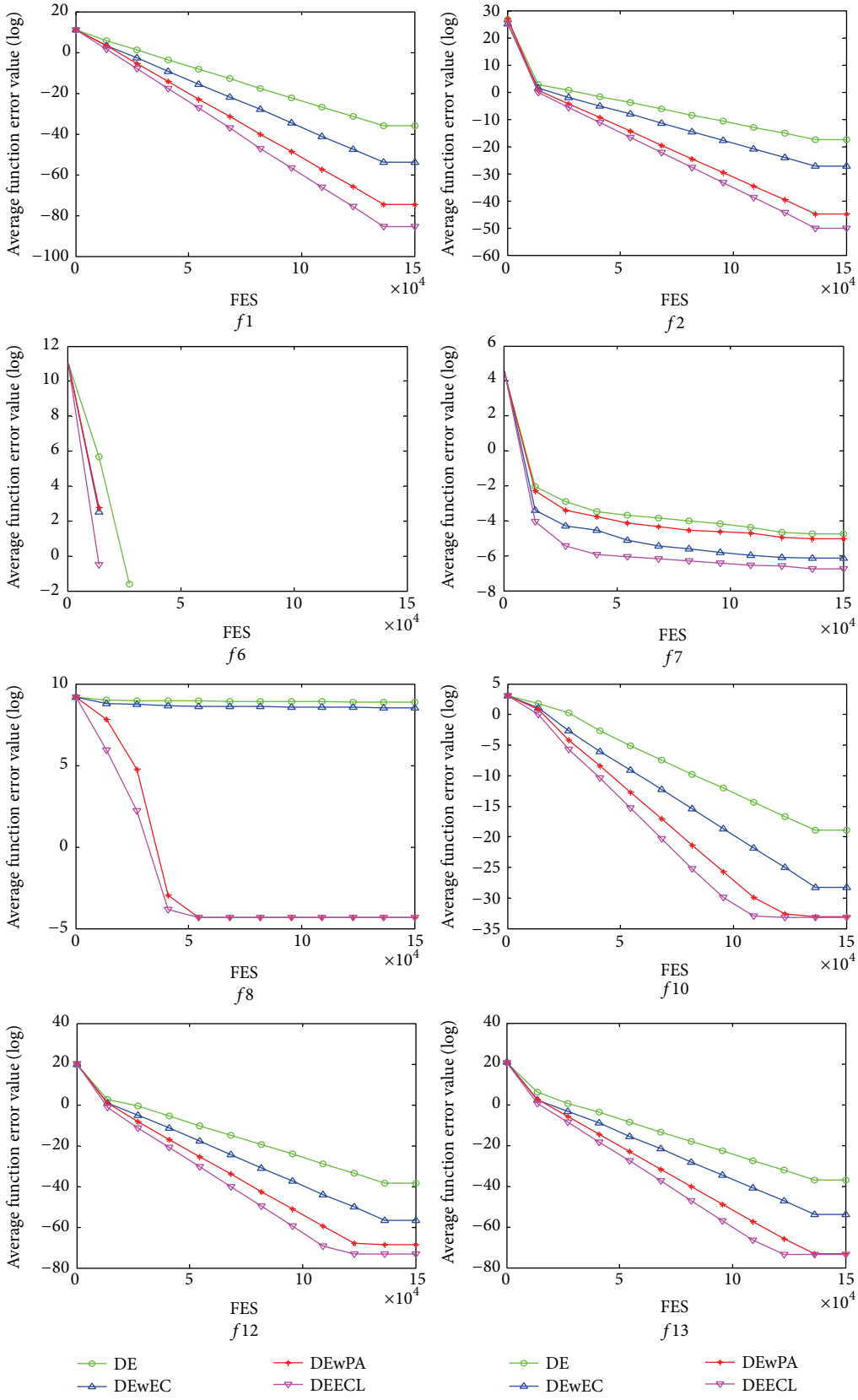


Figure 1: Evolution of the average function error values derived from DE, DEwEC, DEwPA, and DEECL versus the number of FES.

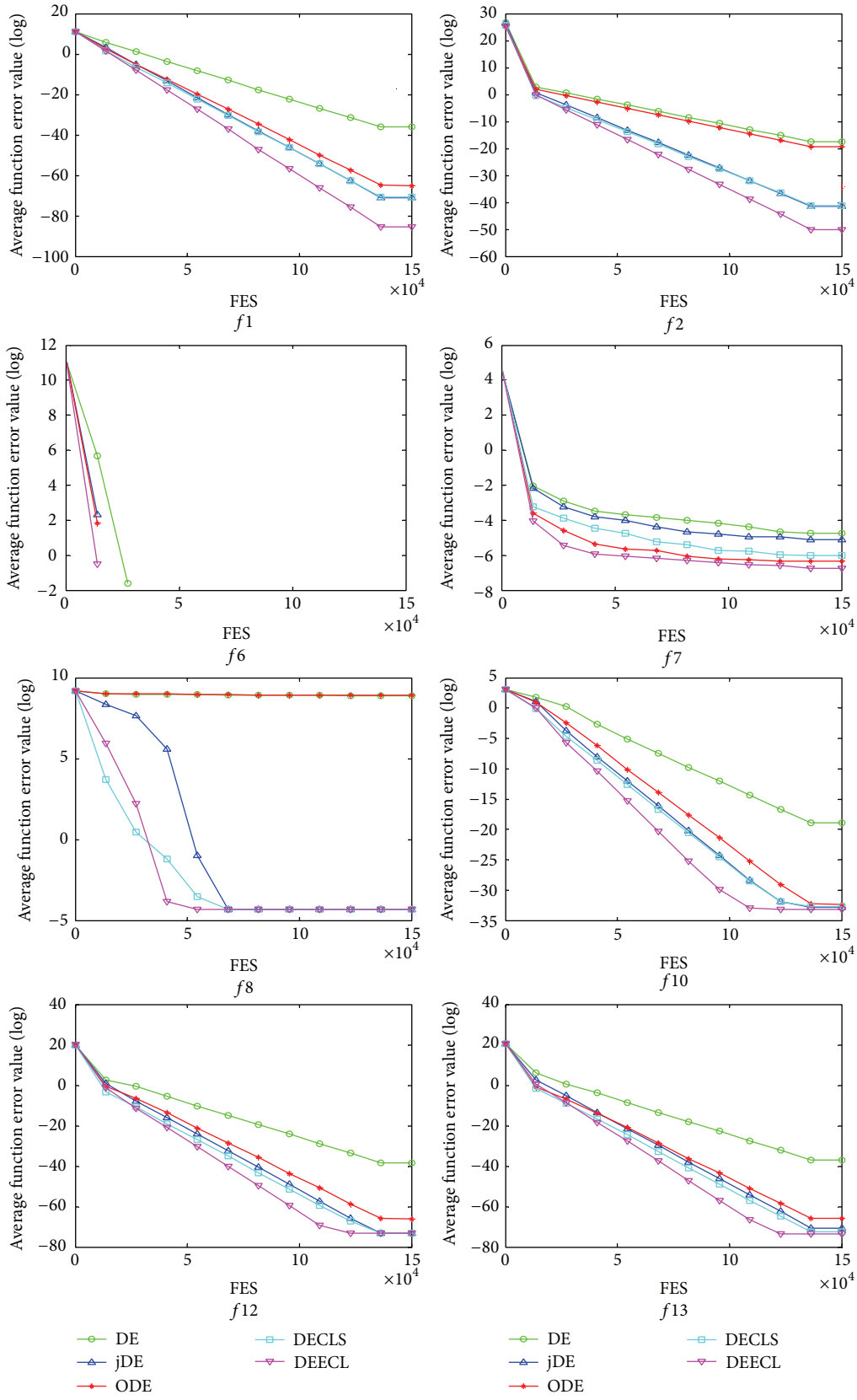


Figure 2: Evolution of the average function error values derived from DE, jDE, ODE, DECLS, and DEECL versus the number of FES.

Table 3: Experimental results of DE, jDE, ODE, DECLS, and DEECL over 30 independent runs for the 13 test functions.

Function	DE Mean ± Std Dev	jDE Mean ± Std Dev	ODE Mean ± Std Dev	DECLS Mean ± Std Dev	DEECL Mean ± Std Dev
$f1$	$2.23E-16 \pm 2.50E-16+$	$1.51E-31 \pm 1.82E-31+$	$6.20E-29 \pm 3.92E-29+$	$2.01E-31 \pm 2.22E-31+$	**6.89E − 38 ± 6.06E − 38**
$f2$	$2.86E-08 \pm 1.26E-08+$	$9.13E-19 \pm 3.70E-19+$	$4.31E-09 \pm 2.61E-09+$	$1.46E-18 \pm 9.36E-19+$	**1.74E − 22 ± 1.21E − 22**
$f3$	$1.88E-01 \pm 6.12E-02+$	$1.85E-02 \pm 6.45E-03\approx$	$1.45E-01 \pm 1.17E-01+$	**5.39E − 04 ± 5.78E − 04−**	$2.42E-02 \pm 3.44E-02$
$f4$	$1.70E-01 \pm 2.13E-01+$	$3.46E-04 \pm 1.23E-04+$	**1.14E − 07 ± 3.43E − 07−**	$3.31E-05 \pm 2.00E-05\approx$	$4.06E-05 \pm 3.05E-05$
$f5$	$1.39E+01 \pm 8.74E-01-$	$1.87E+01 \pm 5.47E-01-$	$2.29E+01 \pm 1.28E+00-$	**5.50E − 05 ± 1.45E − 04−**	$2.95E+01 \pm 2.21E+01$
$f6$	**0.00E + 00 ± 0.00E + 00≈**	**0.00E + 00 ± 0.00E + 00≈**	**0.00E + 00 ± 0.00E + 00≈**	**0.00E + 00 ± 0.00E + 00≈**	**0.00E + 00 ± 0.00E + 00**
$f7$	$8.82E-03 \pm 2.61E-03+$	$5.89E-03 \pm 1.45E-03+$	$1.78E-03 \pm 6.21E-04+$	$2.45E-03 \pm 2.36E-03+$	**1.17E − 03 ± 6.52E − 04**
$f8$	$7.31E+03 \pm 3.75E+02+$	**1.34E − 02 ± 1.82E − 12≈**	$7.51E+03 \pm 2.36E+02+$	**1.34E − 02 ± 1.82E − 12≈**	**1.34E − 02 ± 1.19E − 12**
$f9$	$1.77E+02 \pm 1.10E+01+$	**0.00E + 00 ± 0.00E + 00≈**	$7.83E+01 \pm 2.21E+01+$	**0.00E + 00 ± 0.00E + 00≈**	**0.00E + 00 ± 0.00E + 00**
$f10$	$5.93E-09 \pm 3.10E-09+$	$5.42E-15 \pm 1.74E-15+$	$8.97E-15 \pm 1.74E-15+$	$6.48E-15 \pm 1.63E-15+$	**4.00E − 15 ± 0.00E + 00**
$f11$	$6.33E-16 \pm 1.16E-15+$	**0.00E + 00 ± 0.00E + 00≈**	**0.00E + 00 ± 0.00E + 00≈**	**0.00E + 00 ± 0.00E + 00≈**	**0.00E + 00 ± 0.00E + 00**
$f12$	$2.20E-17 \pm 1.81E-17+$	$1.97E-32 \pm 8.15E-33+$	$2.23E-29 \pm 2.32E-29+$	$1.64E-32 \pm 1.55E-33+$	**1.57E − 32 ± 2.74E − 48**
$f13$	$8.26E-17 \pm 3.59E-17+$	$2.09E-31 \pm 2.93E-31+$	$2.57E-29 \pm 3.07E-29+$	$3.25E-32 \pm 2.37E-32+$	**1.36E − 32 ± 3.70E − 34**
−	1	1	2	2	
+	11	7	9	6	
≈	1	5	2	5	

Table 4: Average rankings of the five algorithms for the 13 test functions achieved by Friedman test.

Algorithm	Ranking
DEECL	**2.04**
DECLS	2.27
jDE	2.65
ODE	3.50
DE	4.54

from the elite individuals to promote the exploitation power and employs a simple and effective parameter adaptation mechanism to enhance the robustness. In the experiments, we use 13 classical test functions that are widely used in the evolutionary computation community to evaluate the performance of DEECL. The experimental results show that DEECL can outperform the conventional DE, jDE, ODE, and DECLS on the majority of the test functions.

In the future, we will apply DEECL to handle more complex optimization problems, such as high-dimensional optimization problems and multiobjective optimization problems.

Conflict of Interests

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

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (nos. 61364025, 61462036, and 41261093), by the Natural Science Foundation of Jiangxi, China (nos. 20151BAB217010 and 20151BAB201015), by the Education Department Youth Scientific Research Foundation of Jiangxi Province, China (nos. GJJ14456 and GJJ13378).

References

[1] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[2] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2011.

[3] J. Vesterstrøm and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *Proceedings of the Congress on Evolutionary Computation (CEC '04)*, vol. 2, pp. 1980–1987, IEEE, June 2004.

[4] L. Wang and L.-P. Li, "Fixed-structure H_∞ controller synthesis based on differential evolution with level comparison," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 120–129, 2011.

[5] F. Martín, L. Moreno, M. L. Muñoz, and D. Blanco, "Initial population size estimation for a differential-evolution-based global localization filter," *International Journal of Robotics and Automation*, vol. 29, no. 3, 2014.

[6] F. Neri and V. Tirronen, "Recent advances in differential evolution: a survey and experimental analysis," *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 61–106, 2010.

[7] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[8] W. Gong, Z. Cai, C. X. Ling, and C. Li, "Enhanced differential evolution with adaptive strategies for numerical optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 41, no. 2, pp. 397–413, 2011.

[9] H. Wang, S. Rahnamayan, H. Sun, and M. G. H. Omran, "Gaussian bare-bones differential evolution," *IEEE Transactions on Cybernetics*, vol. 43, no. 2, pp. 634–647, 2013.

[10] A. Deb, J. S. Roy, and B. Gupta, "Performance comparison of differential evolution, particle swarm optimization and genetic algorithm in the design of circularly polarized microstrip antennas," *IEEE Transactions on Antennas and Propagation*, vol. 62, no. 8, pp. 3920–3928, 2014.

[11] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[12] J. Zhang and A. C. Sanderson, "Jade: adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[13] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Applied Soft Computing Journal*, vol. 11, no. 2, pp. 1679–1696, 2011.

[14] R. S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 64–79, 2008.

[15] H. Wang, Z. Wu, and S. Rahnamayan, "Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems," *Soft Computing*, vol. 15, no. 11, pp. 2127–2140, 2011.

[16] D. Jia, G. Zheng, and M. Khurram Khan, "An effective memetic differential evolution algorithm based on chaotic local search," *Information Sciences*, vol. 181, no. 15, pp. 3175–3187, 2011.

[17] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 55–66, 2011.

[18] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[19] N. Noman and H. Iba, "Accelerating differential evolution using an adaptive local search," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 107–125, 2008.

[20] Q.-K. Pan, P. N. Suganthan, L. Wang, L. Gao, and R. Mallipeddi, "A differential evolution algorithm with self-adapting strategy and control parameters," *Computers and Operations Research*, vol. 38, no. 1, pp. 394–408, 2011.

[21] D. Kranjcic and G. Stumberger, "Differential evolution-based identification of the nonlinear kaplan turbine model," *IEEE Transactions on Energy Conversion*, vol. 29, no. 1, pp. 178–187, 2014.

[22] Q.-K. Pan, L. Wang, L. Gao, and W. D. Li, "An effective hybrid discrete differential evolution algorithm for the flow shop scheduling with intermediate buffers," *Information Sciences*, vol. 181, no. 3, pp. 668–685, 2011.

[23] L. X. Tang, Y. Zhao, and J. Y. Liu, "An improved differential evolution algorithm for practical dynamic scheduling in steelmaking-continuous casting production," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 2, pp. 209–225, 2014.

[24] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 3, pp. 526–553, 2009.

[25] M. G. Epitropakis, D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis, "Enhancing differential evolution utilizing proximity-based mutation operators," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 99–119, 2011.

[26] S. Kundu, S. Das, A. V. Vasilakos, and S. Biswas, "A modified differential evolution-based combined routing and sleep scheduling scheme for lifetime maximization of wireless sensor networks," *Soft Computing*, vol. 19, no. 3, pp. 637–659, 2014.

[27] T. Bhadra and S. Bandyopadhyay, "Unsupervised feature selection using an improved version of differential evolution," *Expert Systems with Applications*, vol. 42, no. 8, pp. 4042–4053, 2015.

[28] Y. Wang, Z. Cai, and Q. Zhang, "Enhancing the search ability of differential evolution through orthogonal crossover," *Information Sciences*, vol. 185, no. 1, pp. 153–177, 2012.

[29] B. Alatas, "Chaotic bee colony algorithms for global numerical optimization," *Expert Systems with Applications*, vol. 37, no. 8, pp. 5682–5687, 2010.

[30] B. Li and W. S. Jiang, "Optimizing complex functions by chaos search," *Cybernetics & Systems*, vol. 29, no. 4, pp. 409–419, 1998.

[31] Y. He, Q. Xu, S. Yang, and L. Liao, "Reservoir flood control operation based on chaotic particle swarm optimization algorithm," *Applied Mathematical Modelling*, vol. 38, no. 17, pp. 4480–4492, 2014.

[32] B. Liu, L. Wang, Y.-H. Jin, F. Tang, and D.-X. Huang, "Improved particle swarm optimization combined with chaos," *Chaos, Solitons & Fractals*, vol. 25, no. 5, pp. 1261–1271, 2005.

[33] T. Xiang, X. Liao, and K.-W. Wong, "An improved particle swarm optimization algorithm combined with piecewise linear chaotic map," *Applied Mathematics and Computation*, vol. 190, no. 2, pp. 1637–1645, 2007.

[34] X. F. Yan, D. Z. Chen, and S. X. Hu, "Chaos-genetic algorithms for optimizing the operating conditions based on RBF-PLS model," *Computers & Chemical Engineering*, vol. 27, no. 10, pp. 1393–1404, 2003.

[35] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.

[36] Y.-W. Shang and Y.-H. Qiu, "A note on the extended Rosenbrock function," *Evolutionary Computation*, vol. 14, no. 1, pp. 119–126, 2006.

[37] S. Garcia and F. Herrera, "An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons," *Journal of Machine Learning Research*, vol. 9, pp. 2677–2694, 2008.

[38] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization," *Journal of Heuristics*, vol. 15, no. 6, pp. 617–644, 2009.

[39] H. Wang, H. Sun, C. Li, S. Rahnamayan, and J.-S. Pan, "Diversity enhanced particle swarm optimization with neighborhood search," *Information Sciences*, vol. 223, pp. 119–135, 2013.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 642429, 15 pages
http://dx.doi.org/10.1155/2015/642429

Research Article

Fusing Swarm Intelligence and Self-Assembly for Optimizing Echo State Networks

Charles E. Martin[1] and James A. Reggia[2]

[1]*HRL Laboratories, LLC, 3011 Malibu Canyon Road, Malibu, CA 90265, USA*
[2]*Department of Computer Science, University of Maryland, College Park, MD 20742, USA*

Correspondence should be addressed to Charles E. Martin; martince444@yahoo.com

Received 26 December 2014; Revised 17 April 2015; Accepted 11 May 2015

Academic Editor: Okyay Kaynak

Optimizing a neural network's topology is a difficult problem for at least two reasons: the topology space is discrete, and the quality of any given topology must be assessed by assigning many different sets of weights to its connections. These two characteristics tend to cause very "rough." objective functions. Here we demonstrate how self-assembly (SA) and particle swarm optimization (PSO) can be integrated to provide a novel and effective means of *concurrently* optimizing a neural network's weights and topology. Combining SA and PSO addresses two key challenges. First, it creates a more integrated representation of neural network weights and topology so that we have just a single, continuous search domain that permits "smoother" objective functions. Second, it extends the traditional focus of self-assembly, from the growth of predefined *target structures*, to functional self-assembly, in which growth is driven by optimality criteria defined in terms of the performance of emerging structures on predefined *computational problems*. Our model incorporates a new way of viewing PSO that involves a population of growing, interacting networks, as opposed to particles. The effectiveness of our method for optimizing echo state network weights and topologies is demonstrated through its performance on a number of challenging benchmark problems.

1. Introduction

In this paper we demonstrate how two very different nature-inspired methodologies, *self-assembly* (SA) [1] and *particle swarm optimization* (PSO) [2], can be integrated to provide a novel and effective means of concurrently optimizing a neural network's weights and topology. Such an approach addresses two important challenges. The first challenge is finding a more integrated representation of neural network weights and topology, so that rather than having to search in both a continuous weight space and a discrete topology space, there is just a single, continuous search domain that permits "smoother" objective functions. The second challenge is extending the traditional focus of self-assembly research from the growth of predefined *target structures* to functional self-assembly, in which growth is driven by optimality criteria defined in terms of the quality or performance of the emerging structures on predefined *computational problems*.

Swarm intelligence systems, which consist of autonomous agents interacting in a simple and local manner, exhibit complex global behavior that emerges as a consequence of local interactions among the agents [3–10]. Researchers have created a wide range of new problem-solving algorithms inspired by nature and based on the governing principles of swarm intelligence [11–17]. Of particular importance to the research presented in this paper is the powerful and broadly applicable swarm intelligence based optimization method known as particle swarm optimization (PSO) [18–23].

Particle swarm optimization has been applied extensively to train neural network weights. A wide variety of different adaptations and hybridizations of PSO have been developed for this purpose [24–30]. Despite the success in using PSO to optimize network weights, there has been only limited success in applying it to topology optimization, and such applications have largely been restricted to feedforward networks. The methods that do exist implement fairly complicated adaptations of the basic PSO algorithm or enforce stringent restrictions on the domain of feasible network topologies [31–33]. While the method presented

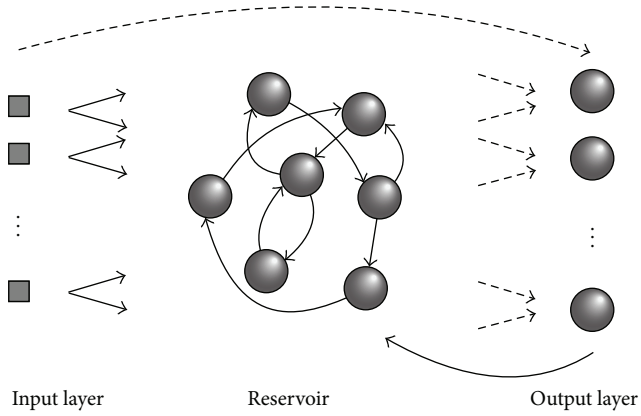


Figure 1: Schematic of an echo state network (ESN). The solid arrows represent connections with fixed weights, and the dashed arrows represent connections with trainable weights. Connections from the input layer to the output layer and from the output layer to the reservoir are optional. As usual, input to the network enters through the input layer and the network's output is generated in the output layer. The bias node and its connections with fixed weights to the reservoir are not shown.

in this paper does not adapt the number of nodes in the network, as some other algorithms do, it optimizes over general, recurrent neural network topologies using one of the most basic forms of PSO, which is a unique feature of our approach. Furthermore, the form of the underlying model of network growth that we present here does not place any constraints on the type of PSO used to drive the optimization, and therefore the user may readily swap in whatever version of PSO is deemed most suitable for the learning task at hand. Our method's incorporation of basic PSO would be particularly advantageous in cases where the topology of a physical network was being optimized.

The work presented here involves the optimization of a relatively recently developed class of recurrent neural network models known as *echo state networks* (ESNs) [34]. Echo state networks have already been successfully applied to a wide range of different problems [35–45]. They consist of an input layer, a hidden layer or "reservoir," and an output layer (shown in Figure 1). Typically, each neuron in the input layer connects to every neuron in the reservoir; there is randomly generated sparse connectivity among reservoir neurons; each neuron in the reservoir connects to each neuron in the output layer; a bias neuron may connect to the neurons in the reservoir, and sometimes connections from the input layer to the output layer and from the output layer to the reservoir are present. The central innovation of the echo state network approach is that only the weights on the connections from the reservoir to the output neurons (output weights) are trained, and the activation functions of the output neurons are linear so all that is needed to train them is linear regression. The remaining weights are typically assigned random values. For an echo state network with N_r reservoir neurons and N_o output neurons, the output weights are trained as follows. A sequence of training data of length $L + M$ is chosen, where $M > N_r$. The first L values of the sequence are passed through the network in order to remove the effects of the initial state of the reservoir. Then, the remaining M values are input into the network, and the resulting reservoir states $\vec{e}_i \in \mathbb{R}^{N_r}$, for $i = 1, \ldots, M$, are assigned to the rows of the matrix $\mathbf{S} \in \mathbb{R}^{M \times N_r}$. For each network input, and resulting reservoir state $\vec{e}_i$, there is a target network output $\vec{d}_i$. The target network outputs are assigned to the rows of the matrix $\mathbf{D} \in \mathbb{R}^{M \times N_o}$ such that the ith rows of $\mathbf{S}$ and $\mathbf{D}$ are the corresponding reservoir state and target output pair. Let $\mathbf{W} \in \mathbb{R}^{N_r \times N_o}$ be the output weight matrix, where the jth column of $\mathbf{W}$ represents the weights on the connections from the reservoir to the jth output neuron. Training the output weights amounts to finding an approximate solution $\mathbf{W}_a$ to the overdetermined system

$$\mathbf{SW} = \mathbf{D}. \tag{1}$$

The output weights $\mathbf{W}_a$ are determined by solving (1) in a "least squares" sense.

Self-assembly involves the self-organization of discrete components into a physical structure, for example, the growth of connections between nodes in a physical network. Work in this area has traditionally focused on what we will refer to as the *classic self-assembly problem*, which entails the design of local control mechanisms that enable a set of components to self-organize into a given target structure, without individually preassigned component positions or central control mechanisms. Issues surrounding self-assembly have been a very active research area in swarm intelligence over the last several years, with recent work spanning computer simulations [46–51], physical robotics [52–57], and the modeling of natural systems [1, 12]. However, to the best of our knowledge, there has been no work on using swarm intelligence methods to extend the classic self-assembly problem to *functional self-assembly*, in which components self-organize into a computing structure that is optimized for a predefined computational problem.

The research presented in this paper is concerned with the self-assembly of neural network architectures. Unlike most past work on self-assembly, a major aspect of the work presented here involves the growth of connections between discrete, spatially separated nodes/components [58]. We recently demonstrated how swarm intelligence in the form of collective movements can increase the robustness of network self-assembly and enhance the ability to grow large, topologically complex neural networks [50]. However, this earlier work focused on the classic self-assembly problem where a target network was prespecified, which is in contrast to the more difficult problem of functional self-assembly that we consider here.

Other related past works in computer science and engineering, where research on artificial neural networks is concerned with application-related performance, have largely ignored the issues of neural network growth, development, and self-assembly, with two exceptions. First, a number of computational techniques have been created to optimize neural network architectures by adding/deleting nodes/connections dynamically during learning [59–63].

Unlike the approach taken here, these past network construction methods do not involve growth or self-assembly in a physical space and so are not considered further. Second, a technique known as *developmental encoding* has been used by researchers evolving neural network architectures with genetic algorithms/programming [64–70]. Unlike the work presented here, in these past techniques different individuals within a population do not directly interact with one another during the development process. Such interaction occurs only indirectly through the processes of selection and crossover.

Both weights and topology affect a neural network's performance. To date, substantially more focus has been placed on techniques for optimizing a neural network's weights as opposed to its topology (the number and connectedness of its nodes). One of the primary reasons for this discrepancy is that the space searched by an optimization method for a good set of weights (the "weight space") is continuous. Thus a good set of weights can be found using one of a wide variety of powerful and well-studied optimization techniques based on local gradient information [71]. Additionally global optimization techniques such as particle swarm optimization (PSO), evolutionary computation (EC), and simulated annealing have proven to be very effective at weight optimization.

When optimizing in continuous domains, specifically multidimensional Euclidean space $\mathbb{R}^n$, optimization algorithms tend to operate under the basic heuristic (referred to here as the continuity-heuristic) that, given two points in a search space, each of which represents a good solution, it is likely that a better solution exists somewhere between or around these points. This heuristic has been found to be generally useful in optimizing objective functions defined on $\mathbb{R}^n$. The "topology space," however, is a discrete space. The discrete nature of this search domain coupled with the intrinsic interdependence between neural network weights (parameters) and topology (structure) results in a variety of additional challenges not encountered when optimizing the weights of a network with a fixed architecture. First, it is common for many of the nodes in a neural network to be identical from a computational perspective, such as the nodes in the hidden layer, which means that many pairs of points in the topology space that are far apart will represent identical or very similar topologies. Second, certain connections may influence performance much more than others. This effect depends on factors such as a network's topology, the learning algorithm used, and the computational problem a network is tasked with solving. This means that in typical representations of the topology space, where a network that has an input-to-output connection would have a nearly identical representation to one that does not have such a connection (all other connections being the same), many points (topologies) that are near each other will represent network architectures that are associated with vastly different fitness values. Third, the quality of a particular topology is dependent on the set of weights associated with it and vice versa. This interdependence means that, rather than having a fixed fitness value, a point (topology) in the topology space has a distribution of fitness values generated by associating different sets of weights with its connections. This fact increases the "roughness" of the fitness landscape defined over the topology space. The first of the above characteristics implies that the distance between two points in the topology space often does not accurately reflect the similarity/dissimilarity of the topologies represented by the points. The second and third characteristics, coupled with the discrete nature of the topology space, imply that nearby points often represent topologies with very different fitness values, which produces very rough fitness/objective functions. Therefore, the properties that make the continuity-heuristic useful in the weight space are largely absent from the topology space.

If we could find a more integrated means of representing neural network weights and topology, such that the search domain consisted of a single, continuous "weight-topology space," then this representation might preserve the continuity-heuristic and permit smoother objective functions. This is precisely what we have done. The integration involves *representing weights and topology using self-assembling neural networks that grow through a single, continuous three-dimensional space*. Our approach makes use of the fact that given two neural networks with different topologies, if the connections that these networks do *not* have in common have weights that are small enough in magnitude, then the networks will have roughly the same *effective* topologies in the sense that signals transmitted via these connections will be highly attenuated and thus tend to have very little influence on network dynamics. Therefore, from the perspective of network performance, it is as if the connections are not actually there. The idea of a neural network having an effective topology is a key concept in the work presented here. It explains why our approach can simultaneously optimize both weights and topology while operating over a continuous space. Specifically, as long as the weight threshold that triggers the removal (addition) of a connection is small enough, then traversing this threshold in the direction that results in the removal (addition) of a connection will yield a network with a new topology but which has nearly identical dynamics compared to its counterpart network. This is the case because the removed (added) connection would have a weight with magnitude near 0. Thus, if the objective function depends only on network dynamics, then this new network and its counterpart will evaluate to nearly identical values under the objective function.

The primary contributions of the work presented here arise from addressing the challenges inherent in the simultaneous optimization of neural network weights and topology. The first contribution is the development of a means of using network self-assembly as a representation of the search domain encountered in this optimization problem, thereby simplifying the domain to a single, continuous space. Second, our work puts forward a new way of viewing PSO that involves a population of growing, interacting networks, as opposed to particles. This adaptation is used to turn the self-assembly process into an optimization process and, to the best of our knowledge, is the first demonstration of such a complimentary relationship between self-assembly and particle swarm optimization. Third, the version of PSO

that our work incorporates, which is a particularly elegant form of the algorithm, has not been previously used to simultaneously optimize neural network weights and topology. Lastly, we demonstrate the effectiveness of a software-based implementation of the integration of self-assembly and PSO by using it to grow high-quality neural network solutions to a variety of challenging benchmark problems from the domains of time-series forecasting and control.

2. Methods

In this section we introduce our model that represents an extension of the traditional self-assembly problem in that the growth of network structures is based on optimality criteria and not on target structures that are specified *a priori*.

2.1. Integrating Self-Assembly and Particle Swarm Optimization. We now present the details of our model for simultaneously optimizing neural network weights and topology. We call this model SINOSA, which stands for Swarm Intelligent Network Optimization through self-assembly. In this model groups of growth cones that belong to different networks simultaneously grow through the same three-dimensional space. During the growth process the growth cones from different networks interact with one another through a mechanism inspired by particle swarm optimization. Concurrently, the networks receive input derived from a computational problem that they must learn to solve. The combination of this interaction, and the activity run through the networks during the development process, leads to the self-assembly of neural networks with weights and topologies that are optimized for solving the problem at hand. An animation that exemplifies the growth process can be viewed at supplied URL (see Supplementary Material available online at http://dx.doi.org/10.1155/2015/642429).

2.1.1. Objects and Relations. Throughout this section the concrete example of the model illustrated in Figure 2 is referenced for clarification. The SINOSA model consists of a set of cells $\mathbf{C}$ with fixed positions in 3D space that are assigned *a priori*. The cells represent neuron cell bodies. Each cell $c_i \in \mathbf{C}$ has a set $\mathbf{N}_{c_i} \subseteq \mathbf{C}$, which may be empty, of "neighbor" cells that it can connect to, where $i = 1, 2, \ldots, |\mathbf{C}|$. In Figure 2 the three large grey spheres represent cells $\mathbf{C}$, and each cell is allowed to connect to any other cell, including itself. Thus, $\mathbf{N}_{c_i} = \mathbf{C}$, for $i = 1, 2, 3$.

Each growing network consists of the same set of cells $\mathbf{C}$ and a unique set of growth cones that guide the network's axons through the three-dimensional space. Given n simultaneously growing networks, each cell c_i has n sets of growth cones $\mathbf{G}_{ij}$, where $j = 1, 2, \ldots, n$. Any given cell c_i contributes the same number of growth cones to each growing network. That is, for all j and ℓ, $|\mathbf{G}_{ij}| = |\mathbf{G}_{i\ell}|$, ensuring that all of the growth cone neighborhoods (explained below) among the growth cones in $\mathbf{G}_i = \bigcup_j \mathbf{G}_{ij}$ are of the same size. If $\mathbf{N}_{c_i}$ is empty, then so is $\mathbf{G}_{ij}$, for all j. The jth growing network gnet_j consists of the set of cells $\mathbf{C}$ and the set of growth cones $\mathbf{G}_j = \bigcup_i \mathbf{G}_{ij}$ that produce the network's growth. That is, gnet_j is defined by the ordered pair $\langle \mathbf{C}, \mathbf{G}_j \rangle$. Because each growing network consists of the same set of cells $\mathbf{C}$, they all have exactly the same number of growth cones ($|\mathbf{G}_j| = |\mathbf{G}_\ell|$, where $j, \ell = 1, 2, \ldots, n$). In Figure 2 the small circles represent growth cones, and the lines that connect the cells and growth cones are axons. In this case there are $n = 3$ growing networks, each having six growing axons. The growing axons of any particular network are shown in a unique line-style (solid, dotted, or dash-dot). To clarify this, Figure 2(b) shows only the solid-line growing network. Figure 2(c) shows the static network that is derived from the solid-line growing network, which is described in the next section. Figure 2(a) illustrates how all three networks simultaneously grow through the same space and share the same three cells.

Let $c_k \in \mathbf{N}_{c_i}$ be the kth neighbor cell of c_i. Then for each c_k, the cell c_i contributes exactly two growth cones (g^s_{ijk} for $s \in \{\text{“+”}, \text{“−”}\}$) to each of the growing networks $j = 1, 2, \ldots, n$. When $s =$ “+” the growth cone represents positively weighted connections, and when $s =$ “–” the growth cone represents negatively weighted connections. The positive-negative growth cone pair g^+_{ijk} and g^-_{ijk} may establish connections to exactly one target cell, namely, c_k. Based on these relations and since $\mathbf{N}_{c_i} = \mathbf{C}$ for $i = 1, 2, 3$ in Figure 2, each of the cells contributes three positive-negative growth cone pairs to each of the three growing networks. However, for the sake of clarity only 6 of the 18 growing axons per network are shown.

2.1.2. Interpreting Network Growth as Self-Assembly. The SINOSA model grows neural networks that, in their completed form, have fixed connections. Thus, it is necessary to interpret the positions of a growing network's growth cones relative to their target cells so as to map a *growing network* gnet_j, to a *static network* snet_j. In particular, if the positive-negative growth cone pair g^+_{ijk} and g^-_{ijk} from cell c_i and growing network gnet_j are both positioned so as to be able to establish a connection to cell c_k, then the weight on the connection from c_i to c_k in the static network snet_j is the sum of the individual weights contributed by the growth cone pair.

In the SINOSA model the function from the space of growing networks to the space of static networks is designed around the need to create a neural network representation that more closely integrates the concepts of topology and connection weight so that the canonical PSO algorithm can be used to optimize these network characteristics effectively. This function is implemented as follows. Each growth cone is considered to be at the center of its own spherically symmetric "weight field" that is finite in extent, and its corresponding weight has a magnitude that decreases to zero as the distance from the growth cone increases. A growth cone establishes a connection with a target cell if the cell is within the boundary of its weight field; otherwise no connection is created. The spacing between cells is such that no more than one cell can be within a growth cone's weight field at any given time. The weight on the connection is

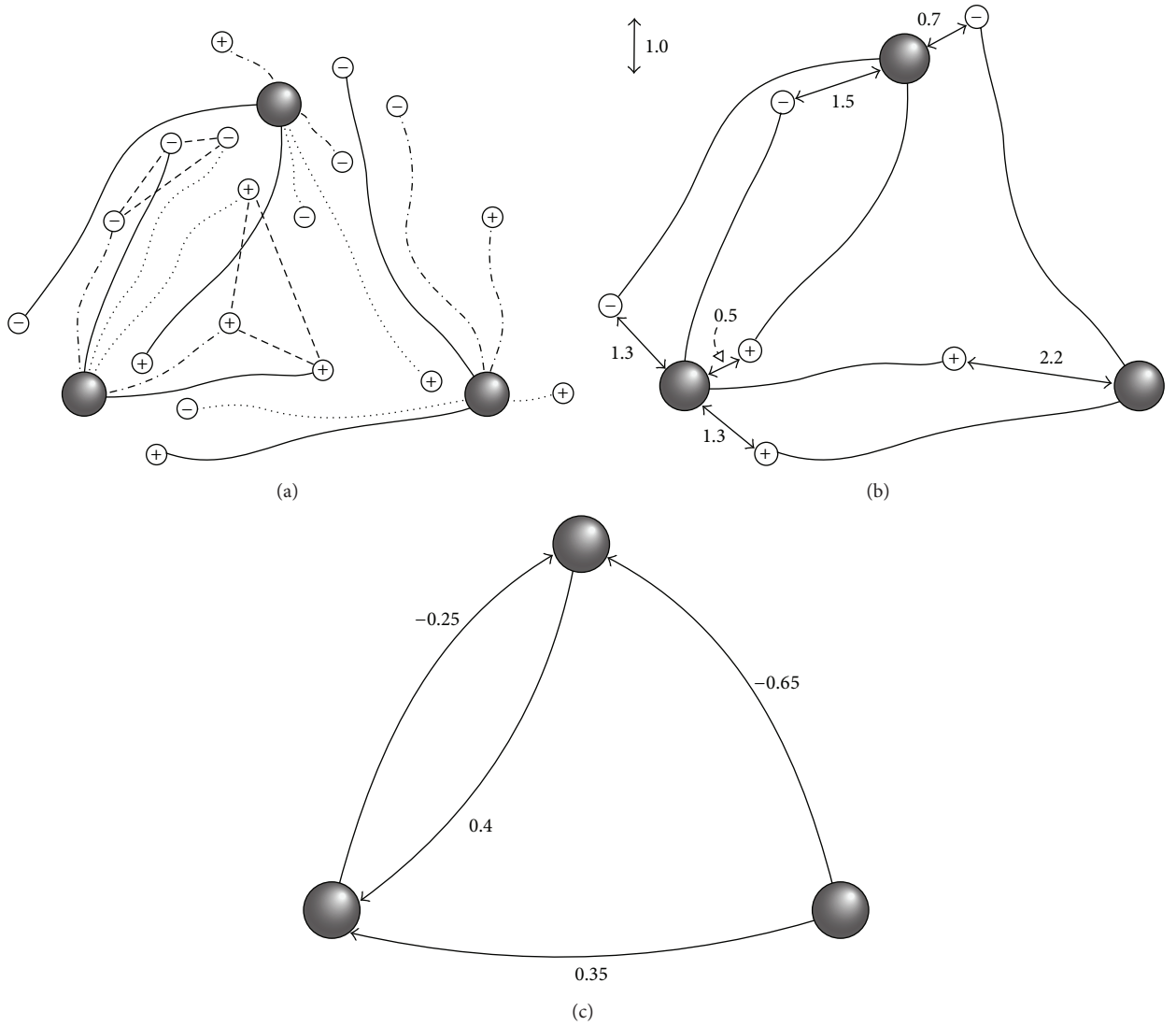


Figure 2: Three growing neural networks and their interpretations as static neural networks based on the SINOSA model. The three large spheres represent cells, the smaller circles represent growth cones, and the lines between cells and growth cones denote connections (axons). The growth cones that are drawn with a "+" symbol have a positive weight field, and those that are drawn with a "−" symbol have a negative weight field. (a) The growth cones and axons that belong to a particular growing network are all shown with the same line-style (solid, dotted, or dash-dot). The straight dashed lines between growth cones indicate two of the six growth cone neighborhoods. Growth cones within a neighborhood interact with one another according to the canonical PSO algorithm. All three growing networks share the same three cells. (b) The solid-line growing network is shown without the other two. (c) The corresponding static network to which the solid-line growing network is mapped based on the proximity of its growth cones to their target cells. The numbers represent connection weights.

the value of the field at the target cell's center. Formally, a *weight field* is a function from $\mathbb{R}$ to $\mathbb{R}$ with the form

$$w(r) = \begin{cases} ar^{\alpha} + b, & \text{if } r < r_0 \\ 0, & \text{if } r \geq r_0, \end{cases} \tag{2}$$

where $a, b \in \mathbb{R}$, $r \geq 0$ is the distance of the target cell from the center of the growth cone, $r_0 > 0$ is the extent of the weight field, and $\alpha > 0$. In the SINOSA model it is assumed that $w(r) \to 0$ as $r \to r_0$. Thus $w(r_0) = ar_0^{\alpha} + b = 0$, which implies that $a = -b/r_0^{\alpha} = -w(0)/r_0^{\alpha}$. Figures 2(b) and 2(c) illustrate how one of the three interacting, growing networks, shown together in Figure 2(a), is mapped to a static network based on the weight field interpretation of the growth cones' positions relative to their target cells. The growth cones that are drawn with a "+" symbol have a positive weight field represented by the function

$$w_{+}(r) = \begin{cases} -\frac{1}{2}r + 1, & \text{if } r < 2 \\ 0, & \text{if } r \geq 2, \end{cases} \tag{3}$$

where r is the distance between a growth cone and one of its target cells. The growth cones that are drawn with a "–" symbol have a negative weight field expressed by the function

$$w_{-}(r) = \begin{cases} \frac{1}{2}r - 1, & \text{if } r < 2 \\ 0, & \text{if } r \geq 2. \end{cases} \quad (4)$$

Thus, in this scenario the weights are restricted to the interval $[-1, 1]$. Figure 2(b) shows the solid-line network's growing axons, along with the distance between each growth cone and its nearest target cell. Figure 2(c) shows the static network derived from the solid-line growing network. The numbers here are the connection weights. This mapping occurs as follows. The cell shown in the lower left hand corner of Figures 2(b) and 2(c) establishes a connection with weight $w_{-}(1.5) = -0.25$ to the upper cell but does not establish a connection with the cell in the lower right hand corner because $w_{+}(2.2) = 0$. The upper cell makes a connection to the lower left hand cell with weight $w_{+}(0.5) + w_{-}(1.3) = 0.4$. The lower right hand cell makes a connection to the lower left hand cell with weight $w_{+}(1.3) = 0.35$, and it makes a connection to the upper cell with weight $w_{-}(0.7) = -0.65$. The other two growing networks are mapped to their corresponding static network representations in an analogous manner.

The function that maps growing networks to static networks is formulated so that a small change in the position of a growth cone produces a small change in the weight on a connection, or if the change in position results in the addition or removal of a connection, then the added or removed connection has a weight that is small in magnitude. In other words, a small change in the physical configuration of a growing network will produce a small change in the weights and topology of the static network to which it is mapped. This characteristic, coupled with the fact that network optimization via the SINOSA model occurs in a single, continuous weight-topology space, results in much smoother objective functions.

2.1.3. Incorporating PSO. Using network self-assembly as our representation scheme yields a single, continuous weight-topology space that needs to be searched during the optimization process. In other words, we need to extend the classic self-assembly problem to functional self-assembly. Given that we have a single, continuous search domain, a wide variety of different optimization algorithms could be used for this purpose. However, we chose to use the PSO algorithm because it is intuitive and effective, and our model incorporates a version of the algorithm for which application to the concurrent optimization of neural network weights and topology has not previously been explored. Specifically, we utilize one of the most basic formulations of PSO, which will be referred to as *canonical* PSO. Canonical PSO specifies that the particles velocities are governed by

$$\vec{v}_i(t+1) \longleftarrow \chi\left(\vec{v}_i(t) + a_p\vec{u}_1 \otimes \left(\vec{p}_{best,i} - \vec{r}_i\right) + a_n\vec{u}_2 \otimes \left(\vec{n}_{best,i} - \vec{r}_i\right)\right), \quad (5)$$

where $\vec{r}_i(t)$ is the position of the ith particle at time t, $\vec{v}_i(t)$ is its velocity, $\vec{p}_{best,i}$ is the current best position of the ith particle, $\vec{n}_{best,i}$ is the best position among any of its neighbor particles, χ is a scaling factor known as the constriction coefficient, a_p and a_n are positive constants, $\vec{u}_1$ and $\vec{u}_2$ are vectors whose components are drawn from the uniform probability distribution over the unit interval, and the $\otimes$ symbol represents the component-wise vector product (i.e., $[a_1 \;\; a_2] \otimes [b_1 \;\; b_2] = [a_1b_1 \;\; a_2b_2]$). It is standard practice to update the positions of the particles using a Forward Euler step with a step-size of 1.0; that is, $\vec{r}_i(t+1) \leftarrow \vec{r}_i(t) + \vec{v}_i(t+1)$. The appeal of this version of PSO lies in its simplicity and in its proven effectiveness on a wide range of optimization problems.

In order to integrate particle swarm optimization and self-assembly the particles in PSO are viewed as being part of a larger structure. Almost all implementations of PSO consider the *particle* to be the fundamental type of object capable of movement and interaction during the optimization process. In the research presented here, the growing *network* plays the role of the fundamental type of object involved in the optimization process. That is, instead of a population of moving particles, there is a population of growing networks. The transition from particles to networks is achieved by having *growth cones* play the role that *particles* do in traditional PSO. Growth cones occur at the leading tips of growing axons (connections) and guide their movements through the physical space. The growth cones' movements are dictated by the canonical PSO equation (5), and because growth cones occur at the leading tips of growing axons, their movements generate network growth. Unlike in traditional PSO, the position of a growth cone (particle), however it is interpreted, is only meaningful when the axon/neuron that it is a part of is taken into account.

Since the growth cones from different growing networks interact with one another according to the canonical PSO algorithm, during the self-assembly process each growth cone must be assigned a quality (fitness) value that indicates the usefulness of the best solution component (connection) the growth cone has found, and it must remember its personal best position, which represents the best connection found by the growth cone up to the current point in the growth process. Specifically, at each discrete time-step $t \in \mathbb{N}$ the performance of each static network $\text{snet}_j(t)$ on some set of training data is determined, where $j = 1, 2, .., n$. For each growing network $\text{gnet}_j(t)$, if the performance of $\text{snet}_j(t)$ is better than the performance of $\text{snet}_j(t - \tau)$ for all $\tau \in \mathbb{N}$ such that $0 < \tau \leq t$, then the fitness value of $\text{gnet}_j(t)$, or more specifically its growth cones $g^s_{ijk} \in \mathbf{G}_j$, is set to the performance value of $\text{snet}_j(t)$, and the personal best position of each growth cone g^s_{ijk} is set to its current position. In theory, it is possible to determine the fitness of each growth cone in a network individually, rather than collectively at the network level. To do this one would need to determine a fitness value proportional to $\mathbb{E}[\text{Network Performance} \mid \text{Growth Cone Weight}]$, the expected value of the network's performance as a function of the growth cone's weight. We chose the former approach for two reasons. First, computing such an expectation value

requires averaging network performance over a very large number of possible growth cone positions that constitute instantiations of different networks. Second, each individual connection has only minimal influence on network performance, and thus optimizing them individually tends to lead to convergence on suboptimal solutions.

Any growth cone g^s_{ijk} must have a set of neighbor growth cones $\mathbf{N}_{g^s_{ijk}}$ that influence its movements. As in most implementations of PSO, the research presented here adheres to the condition that the neighbor relation is symmetric. That is, if $g^s_{i\ell k}$ is a neighbor of g^s_{ijk}, then g^s_{ijk} is a neighbor of $g^s_{i\ell k}$. There is a wide variety of different ways that a growth cone's neighbors could be selected. However, certain characteristics of the self-assembly/optimization process limit the number of useful choices. It is an underlying assumption of the PSO algorithm that the closer two neighbor particles get to one another, the more similar the solutions or solution components that their positions represent are. It is essential for the effectiveness of the PSO algorithm that if two growth cones g^s_{ijk} and $g^s_{i\ell k}$ are neighbors, and they occupy the same position, then they represent exactly the same weighted connection in their respective static networks snet_j and snet_ℓ.

In the SINOSA model, if two growth cones occupy the same position but are guiding axons from different cells, then they represent two completely different connections (solution components). Likewise, if two growth cones occupy the same position but do not have exactly the same set of target cells, then they may represent different connections. These two scenarios, and the need for growth cones that are neighbors to avoid circumstances where they occupy the same position and yet represent different weighted connections, lead to three necessary growth cone neighborhood properties. First, if a pair of growth cones are neighbors, then they must be guiding axons from the same cell. Second, if a pair of growth cones are neighbors, then they must have exactly the same set of target cells. Third, if a pair of growth cones are neighbors, then their weight fields must be expressed by the same function. The following is a simple and effective way of choosing a growth cone's neighbors such that these properties are satisfied. For any cell c_i and growing network gnet_j, the neighbor growth cones of the growth cone $g^s_{ijk} \in \mathbf{G}_{ij}$ with target cell c_k and sign s are members of the set $\mathbf{N}_{g^s_{ijk}} \subset \{g^s_{i\ell k} \in \mathbf{G}_{i\ell} \mid \ell = 1, 2, \ldots, n\}$. In Figure 2(a) the dashed lines between growth cones explicitly show two of the 18 growth cone neighborhoods (only 6 of the 18 growing axons per network are shown). Because each growth cone neighborhood consists of three growth cones connected in a ring topology, $\mathbf{N}_{g^s_{ijk}} = \{g^s_{i\ell k} \in \mathbf{G}_{i\ell} \mid \ell = 1, 2, 3 \wedge \ell \neq j\}$.

When the SINOSA model is used to grow a network that is optimized for a computational problem, on every time-step of the growth process, the performance of each static network is evaluated and used to update the fitness values of the growth cones. The positions of the growth cones are then updated according to the canonical PSO algorithm. Then, on the next time-step, the new physical configurations of the three growing networks are mapped to their corresponding static networks, and the evaluation process repeats. The growth process terminates, and the best performing static network found during the growth process is returned, after a predefined number of time-steps, or once one of the static networks satisfies a prespecified performance criterion.

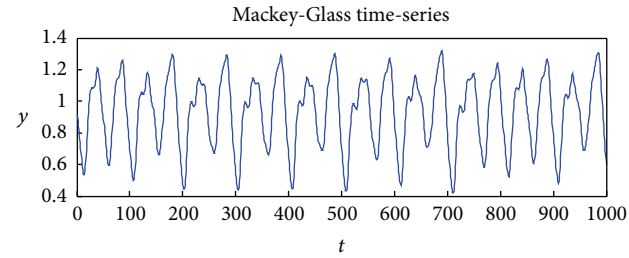


Figure 3: An example of the time-series generated by (6) with parameters $\alpha = 0.2$, $\beta = 10.0$, $\gamma = 0.1$, and $\tau = 17$.

2.2. Experimental Methods. Here, we cover the implementation details of the SINOSA model when it is used to optimize neural networks for the Mackey-Glass time-series forecasting problem and the double pole balancing problem. These problems were selected because they are challenging and widely used benchmark tasks in the domains of time-series forecasting and control, and a wide variety of different neural network training/optimization algorithms have been used in solving them.

2.2.1. Computational Test Problems. The first problem is forecasting the chaotic Mackey-Glass time-series [39, 72, 73]. The time-series is generated by the delay differential equation

$$\frac{dy}{dt} = \frac{\alpha y(t-\tau)}{1 + y(t-\tau)^\beta} - \gamma y(t), \tag{6}$$

where α, β, γ, and $\tau \in \mathbb{R}^+$. When $\tau > 16.8$ the time-series is chaotic. Figure 3 shows a sample of the time-series.

The second problem is the double pole balancing problem (see Figure 4), which is a classic benchmark control problem, particularly for neural network controllers (neural controllers) [74–76]. The double pole balancing problem consists of using a controller to balance two poles with different lengths that are hinged to the top of a cart that moves along a track of finite length. The controller attempts to keep the poles up-right by applying a force F_c to either side of the cart in a direction parallel to the track. To be successful, the controller must keep the cart within a specified distance x_{limit} from the center of the track, and it must keep each pole within a specified angular limit θ_{limit} from the vertical. The equations governing the dynamics of a cart with N poles can be found in [76].

2.2.2. Implementation Details. The SINOSA model is implemented as a simulation environment written in Java. The computational experiments presented in Section 3 each ran on a computer with two quad-core 2.33 GHz Intel Xeon processors, 8 GB of shared RAM, and 12 MB of L2 cache

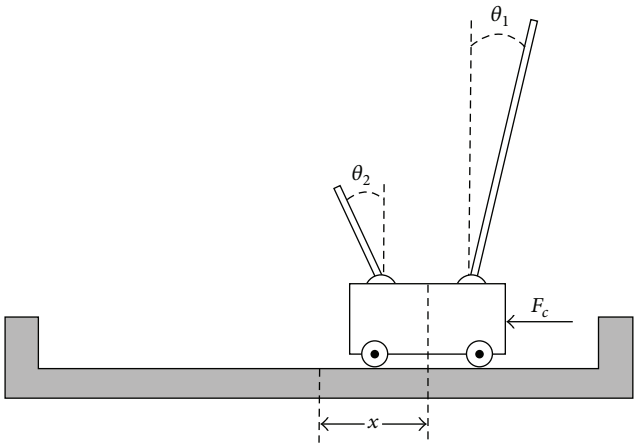


Figure 4: The cart-pole system used in the double pole balancing problem. The state of the system is defined by the position x of the cart relative to the center of the track and the angular positions θ_1 and θ_2 of the large and small poles relative to the vertical. The control force F_c is applied to the side of the cart, in a direction parallel to the track.

per processor. The computational requirements listed here are for the growth of echo state networks using collective movements, unless stated otherwise. The environment in which networks grow was unbounded and no particular units were assigned to time and distance. The components of growing networks (cells, axons, and growth cones) were not able to collide with one another. The cells' positions remained fixed throughout the growth process. Unless stated otherwise, in every experiment the positions of the cells were fixed on a centered rectangular lattice with 8.0 distance-units between adjacent lattice points; there were 16 growing networks, and the growth cone neighborhoods adhered to a von Neumann topology (square lattice with periodic boundary conditions).

The dynamics of the growth cones are governed by the canonical PSO equation (5), where $a_p = a_n = 2.0$, $\chi = 0.65$ for the Mackey-Glass experiments, and $\chi = 0.725$ for the double pole balancing experiments. For all of the experiments, each growth cone's weight field was linear ($\alpha = 1$) and had a radius $r_0 = 2.0$. By convention, one time-step in the SINOSA model is equivalent to 1.0 unit of time.

Many of the experimental results presented in Section 3 are compared to *control cases* that incorporated random growth cone movements, as opposed to collective movements driven by the canonical PSO equation, generated as follows. At every time-step of the growth process each growth cone is placed at a unique, randomly selected position that is less than a distance r_0 from the center of its target cell, where r_0 is the extent of the growth cone's weight field. This way, the weights on the connections are randomly generated.

For each of the computational problems discussed in Section 2.2.1, the SINOSA model was used to grow echo state networks as solutions. The computational experiments described in the Results were designed to test the optimization capabilities of the SINOSA model.

3. Results

3.1. Mackey-Glass Time-Series. In all of the experiments that involve the Mackey-Glass time-series the parameters of (6) were set to the following commonly used values: $\alpha = 0.2$, $\beta = 10$, $\gamma = 0.1$, and $\tau = 17$. These values yield a "mildly" chaotic time-series. The time-series was generated by solving (6) using the Matlab delay differential equation solver *dde23* with a maximum step-size of 1.0, a relative error tolerance of 10^{-4}, and an absolute error tolerance of 10^{-16}. For every time-series generated from (6), an initial sequence of data points was randomly generated from the uniform probability distribution over the interval $[0, 1]$, and (6) was integrated for 1000 time-steps before collection of the time-series data began. This initial run-off period was necessary to remove the effects of the randomly generated initial condition. Consecutive data points in the sequences generated by the Mackey-Glass system were separated by 1.0 units of time. Data from the Mackey-Glass system was made more appropriate for processing by neural networks by mapping it into the interval $[-1, 1]$ using the hyperbolic-tangent function $\tanh(x) = (e^{2x}-1)/(e^{2x}+1)$. Network output was mapped back to the original range using the inverse of $\tanh(x)$ for testing, validation, and analysis. Each reservoir neuron used $f(x) = \tanh(x)$ as its transfer function and had an internal state governed by the leaky integrator differential equation [34]. The output neuron used the linear transfer function, $f(x) = x$, and did not have an internal state.

The ESNs grown for the Mackey-Glass time-series consisted of a single input neuron, a bias neuron, a single output neuron, and 50 neurons in the reservoir. The growth cones were permitted to establish connections from the input neuron to the reservoir neurons, from the bias neuron to the reservoir, and from the reservoir back to the reservoir. Additionally, each reservoir neuron had a permanent connection to the output neuron because the weights on the reservoir-to-output connections were derived using linear regression. The "Echo State Property," which is a necessary condition on the reservoir dynamics for achieving good performance and in past work has typically been attained by manually scaling the spectral radius of the reservoir weight matrix to be less than one [34], was derived entirely through the network growth process.

The input neuron's set of neighbor neurons was the entire reservoir, as was the case for the bias neuron. Each reservoir neuron's set of neighbor neurons consisted of 5 randomly selected neurons in the reservoir. The output neuron did not have any neighbor neurons, because it did not have any growing axons. For each one of the 16 simultaneously growing networks, each neuron contributed one positively weighted growth cone ($b = 1$ in (2)) and one negatively weighted growth cone ($b = -1$), per neighbor neuron. Each of these positive-negative growth cone pairs had the same, single target neuron.

On every time-step each growing network was mapped to the static network represented by its current physical configuration. Before applying input to a network, the internal state and/or output of each neuron was set to zero. For each static network, the weights on the connections from the reservoir

neurons to the output neuron were trained using teacher-forcing [39] with a sequence generated from the Mackey-Glass system that had 2100 data points. The first 100 data points of this sequence were fed into a network prior to any training with the purpose of removing the effects of the initial state of the reservoir. The next 2000 data points were then fed into the network, and linear regression in the form of the least squares method was performed between the resulting reservoir states (activities of the reservoir neurons) and the desired network outputs. The topology and weights of the connections from the input neuron to the reservoir, from the bias neuron to the reservoir, and from the reservoir neurons back to the reservoir were determined by the growth process.

The performance of an ESN on the data used to train the output weights is typically not a good measure of the network's ability to generalize to new data [77]. Thus, during the growth process the generalization performance of each static network was computed on every time-step and used to update the fitness values of the growing networks. The generalization performance measure was the normalized root mean square error computed over a set of 84-step predictions (NRMSE_{84}) [34]. Specifically, on every time-step, after training the output weights, the NRMSE_{84} was computed for each static network on a group of 20 randomly selected sequences from the Mackey-Glass system. Each of these sequences consisted of 184 data points. In order to prevent overgeneralization, every 10 time-steps a new set of 20 sequences was randomly pulled from a pool of 200 different sequences. At the end of the growth process, which lasted for 3600 time-units, the best positions of the growth cones in each growing network represent the best performing networks found over the course of the entire growth process. There is one best performing network per growing network and it is instantiated by translating the best positions of the network's growth cones into the corresponding static network. The performances of the best static networks were validated by computing the NRMSE_{84} of each network using 100 new sequences, each of a length of 2084. The best performing network on this validation data was taken as the solution. Towards the end of a growth process the growing networks tend to converge on a particular configuration and this final validation step ensures that the solution network has the best *generalization* performance.

For the SINOSA model, 37 trials were run with collective movements generated by canonical PSO, and 38 trials were run with random movements. In the Mackey-Glass experiments, on average, one epoch of growth (explained in Section 3.1) of an ESN with 50 neurons in its reservoir requires 2 hours of CPU time. One epoch of growth of an ESN with 100 neurons in its reservoir requires 5 hours of CPU time. One epoch of growth of an ESN with 400 neurons in its reservoir requires 3 days of CPU time. The networks grown using collective movements have a mean NRMSE_{84} that is 68% smaller than those grown with random movements with 95% confidence.

Once the growth process had finished (after 3600 time-units) the grown networks were further optimized by refining the search process and running it for an additional 3600 time-units. This refinement was implemented by continuing the growth (search) process with new growth cones that had weight fields that were smaller in maximum magnitude. Specifically, for each connection in the best performing static network found during the first epoch, except the connections from the reservoir to the output neuron, a fixed connection with the same weight was created between the corresponding cells in the set **C**. When a static network was instantiated from a growing network during the second epoch, the weights on the connections in the static network were the sum of the weight values contributed by the growth cones and the fixed connections.

Table 1: Mean NRMSE_{84} values on the Mackey-Glass time-series for networks grown with the SINOSA model.

Epoch	Collective movements	Random movements
1	$5.89 \cdot 10^{-3} \pm 3.3 \cdot 10^{-4}$	$1.84 \cdot 10^{-2} \pm 8 \cdot 10^{-4}$
2	$4.98 \cdot 10^{-3} \pm 3.2 \cdot 10^{-4}$	$1.48 \cdot 10^{-2} \pm 7 \cdot 10^{-4}$

The network growth process generated by the SINOSA model was extended by one epoch, for a total of two epochs of growth. Table 1 compares the results obtained using collective movements, with those obtained using random movements, when the growth process was extended. Each numeric value represents the mean NRMSE_{84} and the 95% confidence interval for the corresponding epoch of growth and class of movements. It can be seen that, for both collective and random movements, there is a small but statistically significant reduction in the mean NRMSE_{84} with each epoch of growth. Furthermore, for each epoch, the mean NRMSE_{84} of the networks grown using collective movements is smaller than that of the networks grown using randomly generated movements.

Further evidence of the effectiveness of the SINOSA approach can be gained through comparison with the studies presented in [34, 39], which represent state-of-the-art performance for Mackey-Glass time-series prediction. In [34], echo state networks with 400 neuron reservoirs were optimized to forecast the Mackey-Glass time-series ($\tau = 17.0$). The best performing of these networks, which was hand-designed by an expert, had an NRMSE_{84} of $2.8 \cdot 10^{-4}$. In [39], using the same parameter values for the Mackey-Glass time-series as used here, an echo state network with a 1000-neuron reservoir was hand-designed by an expert that had an NRMSE_{84} of $6.3 \cdot 10^{-5}$. The SINOSA model was used to grow echo state networks with 400 neurons in their reservoirs to forecast the Mackey-Glass time-series. The grown networks produced an average NRMSE_{84} of $3.86 \cdot 10^{-5}$, and the best of these networks had an NRMSE_{84} of $2.73 \cdot 10^{-5}$. On average, the grown networks with 400 neurons outperformed the best hand-designed 400-neuron ESN by about an order of magnitude, and they also performed better than the 1000-neuron ESN. These results provide strong evidence of the effectiveness of using the SINOSA model to grow echo state networks, as opposed to the standard approach of optimizing them through trial and error.

3.2. Double Pole Balancing Problem. In all of the experiments that dealt with the double pole balancing problem,

the parameters were set to the most commonly used values [75] as follows: mass of the cart $m_c = 1\,\text{kg}$, mass of the 1st pole $m_1 = 0.1\,\text{kg}$, mass of the 2nd pole $m_2 = 0.01\,\text{kg}$, coefficient of friction between the cart and the track $\mu_c = 5 \cdot 10^{-4}\,\text{Ns/m}$, coefficients of friction between the poles and their hinges $\mu_1 = \mu_2 = 2 \cdot 10^{-6}\,\text{Nms}$, length of the 1st pole $l_1 = 0.5\,\text{m}$, and length of the 2nd pole $l_2 = 0.05\,\text{m}$. The control force was restricted to the interval $F_c \in [-10\,\text{N}, 10\,\text{N}]$. The parameters defining the domain of successful control were set to $x_{\text{limit}} = 2.4\,\text{m}$ and $\theta_{\text{limit}} = 36^\circ$. As is the case in most past work, the equations governing the dynamics of the system were solved numerically using a fourth-order Runge-Kutta method with a step-size of 0.01 s. During a simulation, a portion of the state of the cart-pole system was given to a neural controller every 0.02 s, at which point the control force was updated. In the experiments presented here, a neural controller was not given velocity information as input; rather, it only received the current positions of the cart and two poles (x, θ_1, and θ_2). This was done in order to make the task of controlling the cart more difficult. These values were scaled to be in the interval $[-1, 1]$ prior to being input into a neural controller. This was done so that the values were in a range that was more appropriate for processing by neurons with hyperbolic-tangent transfer functions. The network output (control signal), which was in the interval $[-1, 1]$, was multiplied by 10.0 N in order to produce the control force. Reservoir neurons and the output neuron used the hyperbolic-tangent function as their transfer function. None of the neurons had an internal state.

The SINOSA model was used to grow echo state networks as controllers for the double pole balancing problem. These networks had three input neurons, one for each type of information; the network was given regarding the state of the cart-pole system (cart position, position of pole #1, and position of pole #2). The reservoir always consisted of 20 neurons. One output neuron was present, which produced the control signal. No bias neuron was used due to the symmetry of the cart-pole system. The growth cones were permitted to establish connections from the input neurons to the reservoir and from the reservoir neurons back to the reservoir. Additionally, each reservoir neuron had a permanent connection to the output neuron. The weights on the reservoir-to-output connections were fixed and drawn randomly with uniform probability from the interval $[-30, 30]$. The network architecture was otherwise identical to that of the Mackey-Glass network (see third paragraph of Section 3.1) except that in this case each reservoir neuron had only 4 neighbor neurons.

During the growth process the performance of each static network was computed on every time-step. The function f_{pole} was evaluated to determine the performance of the echo state networks grown as controllers for the double pole balancing problem and is given by

$$f_{\text{pole}} = 10^{-4} n_I + 0.9 f_{\text{stable}} + 10^{-5} n_{II} + 30 \frac{n_S}{625}. \quad (7)$$

Equation (7) was introduced in [75] and is based on performance (fitness) functions presented in past works on the double pole balancing problem. To compute the first term in (7) the cart-pole system is set to the initial state ($\theta_1(0) = 4.5^\circ, \dot{\theta}_1(0) = \theta_2(0) = \dot{\theta}_2(0) = x(0) = \dot{x}(0) = 0$). The network is then allowed to control the system for up to 1,000 time-steps. The number of time-steps n_I that the controller keeps the cart and poles in the success domain ($x \in [-2.4\,\text{m}, 2.4\,\text{m}]$ and $\theta_1, \theta_2 \in [-36^\circ, 36^\circ]$) is counted. If the system leaves the success domain at any time prior to time-step 1,000, then the simulation stops. The second term is a measure of the stability of the system during the last 100 time-steps while under neural network control and is expressed by the function

$$f_{\text{stable}} = \begin{cases} 0, & n_I < 100 \\ \dfrac{0.75}{\sum_{i=n_I-100}^{n_I} \rho_i}, & n_I \geq 100, \end{cases} \quad (8)$$

where

$$\rho_i = \left(|x(i)| + |\dot{x}(i)| + |\theta_1(i)| + |\dot{\theta}_1(i)| \right). \quad (9)$$

The third and fourth terms are measures of a neural controller's ability to generalize. If $n_I = 1000$ after computing the first term, then the neural controller is allowed to control the system for up to additional 100,000 time-steps. The number of additional time-steps n_{II} that the controller keeps the cart and poles in the success domain is counted, and the simulation stops if the system leaves the success domain or $n_{II} = 100,000$. The fourth term is computed by putting the cart-pole system in 625 different initial conditions and allowing the network to control it for up to 1,000 time-steps from each starting configuration. The variable n_S represents the number of different initial conditions from which the neural controller was able to keep the system in the success domain for 1,000 consecutive time-steps. The 625 unique initial conditions are defined in [74].

On every time-step of the growth process each growing network was mapped to the static network represented by its current physical configuration so that its performance could be computed by evaluating (7). Before applying input to a network the output of each neuron was always set to zero. Before a network was permitted to control the cart and poles the dynamics of the cart-pole system were evolved for 0.2 s, and the resulting sequences of 10 system states were input into the network. The neural network growth process lasted for 600 time-units, after which the static network with the best performance (largest value of f_{pole}) was taken as the solution.

A total of 51 trials were run starting from different, randomly generated initial conditions. In the double pole balancing experiments 200 time-steps of growth require approximately 0.3 hours of CPU time, 400 time-steps require 1.4 hours, and 600 time-steps require 3.3 hours. Table 2 compares the performance of networks grown using collective movements to the performance of networks grown using random movements. The comparison of performance is made every 200 time-steps during the growth process. Each of the numeric values in the tables is shown with its 95% confidence interval. The values in Table 2 were computed as follows. For each trial, and at each of the three predefined time-steps (200, 400, and 600), two measures of the best performing network at that point in the growth

Table 2: Performance values on the double pole balancing problem for networks grown with the SINOSA model.

Time-step	Collective, Measure_{II}	Random, Measure_{II}
200	0.667 [0.530, 0.780]	0.026 [0.005, 0.135]
400	0.961 [0.868, 0.989]	0.053 [0.015, 0.173]
600	1.0 [0.930, 1.0]	0.053 [0.015, 0.173]
Time-step	Collective, Measure_{S}	Random, Measure_{S}
200	372 ± 29	10 ± 5
400	462 ± 10	28 ± 15
600	478 ± 7	41 ± 17

process were recorded. The first measure was whether or not the network succeeded in achieving $n_{II} = 100,000$ when computing (7). The second measure was the value of n_S. In Table 2 the term Measure_{II} refers to the fraction of best performing networks that achieved $n_{II} = 100,000$. The term Measure_S refers to the average value of n_S taken over all of the best performing networks. From these results it is clear that the networks grown with collective movements vastly outperform those grown with randomly generated movements on both performances measures.

A study that lends itself to comparison is presented in [75], which represents state-of-the-art performance on the double pole balancing problem. In this case echo state networks were optimized as controllers for the double pole balancing problem via a state-of-the-art form of evolutionary strategies that uses covariance matrix adaptation (CMA-ES). In this study CMA-ES was used to optimize the output weights and the spectral radius of the reservoir weight matrix. The experiments discussed in this section, in which the SINOSA model was used to grow ESNs as controllers for the double pole balancing problem, adhered to the same experimental setup and methods used in [75], except that in our study the grown neural controllers received only 10 inputs from the cart-pole system prior to beginning control instead of 20. Because evaluating the fitness/performance of the networks during the optimization process is the computational bottleneck, the number of such evaluations during an optimization run is a good measure of the overall computational cost of the process. On average it required 19,796 evaluations for the CMA-ES approach to find a neural controller capable of successfully controlling the cart for at least 200 out of the 625 initial configurations (the average was 224), and of these networks 91.4% were able to successfully control the cart for the additional 100,000 time-steps when it was started in the standard initial configuration. These results are very good with respect to past work on the double pole balancing problem. The SINOSA model was able to grow much better performing neural controllers and at much less computational expense. After only 9600 evaluations, on average, the best performing grown networks were able to successfully control the cart for 478 of the 625 initial configurations, and of these networks 100% of them were able to successfully control the cart for the additional 100,000 time-steps.

4. Discussion

The SINOSA model incorporates an integrated representation of a network's weights and topology. The objects in this representation are cells (neurons), axons, and growth cones. The cells have fixed positions, but the growth cones are able to guide the cells' axons through a continuous, three-dimensional space, producing a mature network with fixed connections and weights. As a result of the integrated representation, it is possible to incorporate the simplest, canonical form of PSO into the model for the purpose of simultaneously optimizing network weights and topologies. In effect, the SINOSA model treats the network self-assembly process as an optimization or search process, in which the simultaneous growth of multiple neural networks is driven by their interactions with one another and with problem related network input.

The ability of the SINOSA model to optimize neural networks for computational tasks was tested using two different very challenging and widely used benchmark problems from the domains of time-series forecasting and control. For each of the computational problems the echo state networks grown using collective movements generated via PSO outperformed those grown using randomly generated movements, and in most circumstances the performance gap was very large. Specifically, compared to the networks grown with random movements, those grown using the SINOSA model with collective movements performed 3 times better on the Mackey-Glass time-series forecasting problem and 19 times better and 12 times better on two generalization measures of the double pole balancing problem. Furthermore, the large improvements in network performance gained over random search come at very little additional computational cost because evaluation of network performance is the bottleneck.

Comparison with control cases that involve random search provides a base level of support for the optimization capabilities of the SINOSA model and demonstrates the feasibility of *functional self-assembly* as a means of network optimization. Further evidence of the effectiveness of the model at optimizing networks can be found by comparing the results presented here with studies that involve different methods of optimizing networks for the Mackey-Glass time-series forecasting problem and the double pole balancing problem. For example, the 400-neuron echo state networks grown using the SINOSA model perform nearly an order of magnitude better than the best performing 400-neuron ESN presented in [34] with the Mackey-Glass time-series. Furthermore, they even outperform the 1000-neuron ESN presented in [39] by an average factor of 1.6. As a point of further comparison, the networks grown via the SINOSA approach outperform those in [75] by an average factor of 2.1. Moreover, our ESNs were optimized with an average of 52% less computational expense. These results are also interesting in that they represent one of the comparatively small number of studies where echo state networks have been successfully trained as neural controllers using reinforcement learning.

It is worth pointing out that the SINOSA model can be cast in a more abstract representation by foregoing the self-assembly component. Imagine we are optimizing a network

with M possible weighted connections. Then, according to (2) there are two distinct one-dimensional Euclidean spaces associated with each possible connection. Furthermore, there is a unimodal function $w_+(r)$ that is *nonnegative* and symmetric defined over the first space and a function $w_-(r) = -w_+(r)$ that is defined over the second space. Each one of these spaces would contain a set of particles (growth cones) that was restricted to move within it. Only those particles within a given space would interact according to the PSO algorithm. A network would be created based on the positions of the particles in exactly the same manner described in Section 2.1.2. We chose to integrate self-assembly into our model from a desire to illuminate the processes by which physical networks might optimize their own weights and topology via self-assembly.

5. Conclusions and Future Directions

The concurrent optimization of neural network weights and topology is a challenging task due in large part to the roughness of the objective functions encountered when the search domain consists of both a continuous weight space and a discrete topology space. Through the SINOSA model we have demonstrated how network self-assembly can provide a useful means of representing this search domain in that the representation simplifies the domain to a single, continuous search space over which smoother objective functions can be defined. Furthermore, by using swarm intelligence in the form of collective movements to drive the network growth process, we were able to turn the self-assembly process into an optimization process.

The four primary contributions of our research are as follows:

(i) The SINOSA model constitutes a new and effective means of simultaneously optimizing the weights and topologies of neural networks. The model was used to grow echo state networks that performed substantially better on benchmark problems than networks optimized via random search. More importantly, the grown networks also outperformed the echo state networks presented in two different past studies, one in which the networks were hand-designed by an expert and the other in which they were optimized using a state-of-the-art form of evolutionary strategies (CMA-ES).

(ii) There has been little past work on using PSO for the concurrent optimization of neural network weights and topology. The examples that do exist tend to involve fairly complicated adaptations of the method, significant constraints on permissible topologies, or hybridizations with other classes of methods such as evolutionary algorithms. In contrast, the SINOSA model uses the elegant canonical form of PSO to govern the growth/optimization process.

(iii) In the vast majority of past work on PSO, the particles are embedded in a high dimensional abstract space, such as the domain of a function, they are the fundamental class of "objects" in the space, and the position of a particle represents a solution or solution component to the problem being solved. In contrast, the SINOSA model incorporates a novel way of viewing PSO in which growth cones (particles) are embedded in a continuous, three-dimensional space that is intended to model physical space, and *growing networks* are the fundamental class of objects in the space.

(iv) Most past work on self-assembly has focused on the classic self-assembly problem, which entails the design of local control mechanisms that enable a set of components to self-organize into a *given* target structure. The SINOSA model represents an extension of the classic self-assembly problem to *functional self-assembly*, which includes the self-assembly of network structures with growth driven by optimality criteria defined in terms of the quality or performance of the emerging structures, as opposed to growth directed towards assembling a prespecified target structure.

There are a variety of potential future research directions for the SINOSA model. Here we mention three possibilities. First, it would be useful to extend this work to allow the number of neurons in the physical space to be able to increase or decrease during network assembly depending on the computational requirements of the problem being solved. Inspiration could likely be drawn from the fairly large number of past studies that involve dynamically modifying the number of nodes in a neural network. Second, further studies are needed to determine to what extent the parameters of the SINOSA model are problem dependent, and what values work well on a wide variety of different problems. Lastly, since its inception, the canonical form of particle swarm optimization has undergone a vast array of adaptations and hybridizations. Many of these enhancements could be incorporated into the SINOSA model without having to alter its fundamental constructs.

Conflict of Interests

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

References

[1] G. M. Whitesides and B. Grzybowski, "Self-assembly at all scales," *Science*, vol. 295, no. 5564, pp. 2418–2421, 2002.

[2] A. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, John Wiley & Sons, West Sussex, UK, 2005.

[3] J. Buhl, D. J. T. Sumpter, I. D. Couzin et al., "From disorder to order in marching locusts," *Science*, vol. 312, no. 5778, pp. 1402–1406, 2006.

[4] A. Cavagna and I. Giardina, "Large-scale behaviour in animal groups," *Behavioural Processes*, vol. 84, no. 3, pp. 653–656, 2010.

[5] A. Dussutour, V. Fourcassié, D. Heibing, and J.-L. Deneubourg, "Optimal traffic organization in ants under crowded conditions," *Nature*, vol. 428, no. 6978, pp. 70–73, 2004.

[6] D. Helbing, I. Farkas, and T. Vicsek, "Simulating dynamical features of escape panic," *Nature*, vol. 407, no. 6803, pp. 487–490, 2000.

[7] H. Hildenbrandt, C. Carere, and C. K. Hemelrijk, "Self-organized aerial displays of thousands of starlings: a model," *Behavioral Ecology*, vol. 21, no. 6, pp. 1349–1359, 2010.

[8] H. Kunz and C. K. Hemelrijk, "Artificial fish schools: collective effects of school size, body size, and body form," *Artificial Life*, vol. 9, no. 3, pp. 237–253, 2003.

[9] C. Lett and V. Mirabet, "Modelling the dynamics of animal groups in motion," *South African Journal of Science*, vol. 104, no. 5-6, pp. 192–198, 2008.

[10] R. Winder and J. A. Reggia, "Using distributed partial memories to improve self-organizing collective movements," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, no. 4, pp. 1697–1707, 2004.

[11] S. Bitam, M. Batouche, and E.-G. Talbi, "A survey on bee colony algorithms," in *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing, Workshops and Phd Forum (IPDPSW '10)*, pp. 1–8, IEEE, New York, NY, USA, April 2010.

[12] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, New York, NY, USA, 1999.

[13] M. Dorigo, G. di Caro, and L. M. Gambardella, "Ant algorithms for discrete optimization," *Artificial Life*, vol. 5, no. 2, pp. 137–172, 1999.

[14] C. J. A. B. Filho, F. B. D. L. Neto, A. J. C. C. Lins, A. I. S. Nascimento, and M. P. Lima, "A novel search algorithm based on fish school behavior," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC '08)*, pp. 2646–2651, IEEE, New York, NY, USA, October 2008.

[15] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Applied Soft Computing*, vol. 8, no. 1, pp. 687–697, 2008.

[16] A. Rodríguez and J. A. Reggia, "Extending self-organizing particle systems to problem solving," *Artificial Life*, vol. 10, no. 4, pp. 379–395, 2004.

[17] X.-S. Yang, "Multiobjective firefly algorithm for continuous optimization," *Engineering with Computers*, vol. 29, no. 2, pp. 175–184, 2013.

[18] Z.-L. Gaing, "Particle swarm optimization to solving the economic dispatch considering the generator constraints," *IEEE Transactions on Power Systems*, vol. 18, no. 3, pp. 1187–1195, 2003.

[19] M. Omran, A. P. Engelbrecht, and A. Salman, "Particle swarm optimization method for image clustering," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 19, no. 3, pp. 297–322, 2005.

[20] Y. Rahmat-Samii, D. Gies, and J. Robinson, "Particle swarm optimization (PSO): a novel paradigm for antenna designs," *The Radio Science Bulletin*, vol. 305, pp. 14–22, 2003.

[21] T. Sousa, A. Silva, and A. Neves, "Particle swarm based data mining algorithms for classification tasks," *Parallel Computing*, vol. 30, no. 5-6, pp. 767–783, 2004.

[22] S. Kitagawa and Y. Fukuyama, "Comparison of particle swarm optimizations for optimal operational planning of energy plants," in *Proceedings of the IEEE swarm intelligence symposium (SIS' 03)*, pp. 159–165, IEEE, June 2005.

[23] M. Voss and J. Howland III, "Financial modelling using social programming," in *Proceedings of the IASTED International Conference on Financial Engineering & Applications*, pp. 1–10, ACTA Press, Calgary, Canada, 2003.

[24] F. van den Bergh, "Particle swarm weight initialization in multi-layer perceptron artificial neural networks," in *Proceedings of the International Conference on Artificial Intelligence*, pp. 42–45, CSREA Press, 1999.

[25] X. Cai, N. Zhang, G. K. Venayagamoorthy, and D. C. Wunsch II, "Time series prediction with recurrent neural networks using a hybrid PSO-EA algorithm," in *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN '04)*, pp. 1647–1652, IEEE, July 2004.

[26] A. Engelbrecht and A. Ismail, "Training product unit neural networks," *Stability and Control: Theory and Applications*, vol. 2, pp. 59–74, 1999.

[27] H.-W. Ge, Y.-C. Liang, and M. Marchese, "A modified particle swarm optimization-based dynamic recurrent neural network for identifying and controlling nonlinear systems," *Computers and Structures*, vol. 85, no. 21-22, pp. 1611–1622, 2007.

[28] C.-F. Juang, "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, no. 2, pp. 997–1006, 2004.

[29] S. Mirjalili, S. Z. Mohd Hashim, and H. M. Sardroudi, "Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm," *Applied Mathematics and Computation*, vol. 218, no. 22, pp. 11125–11137, 2012.

[30] J. R. Zhang, T. M. Lok, and M. R. Lyu, "A hybrid particle swarm optimization-back-propagation algorithm for feedforward neural network training," *Applied Mathematics and Computation*, vol. 185, no. 2, pp. 1026–1037, 2007.

[31] M. Carvalho and T. B. Ludermir, "Particle swarm optimization of neural network architectures and weights," in *Proceedings of the 7th International Conference on Hybrid Intelligent Systems (HIS '07)*, pp. 336–339, IEEE, Kaiserlautern, Germany, September 2007.

[32] S. Kiranyaz, T. Ince, A. Yildirim, and M. Gabbouj, "Evolutionary artificial neural networks by multi-dimensional particle swarm optimization," *Neural Networks*, vol. 22, no. 10, pp. 1448–1462, 2009.

[33] C. Zhang and H. Shao, "An ANN's evolved by a new evolutionary system and its application," in *Proceedings of the 39th IEEE Confernce on Decision and Control*, vol. 4, pp. 3562–3563, IEEE, Sydney, Australia, December 2000.

[34] H. Jaeger, "The echo state approach to analysing and training recurrent neural networks," GMD Report 148, Fraunhofer Institute for Autonomous Intelligent Systems (AIS), 2001.

[35] E. A. Antonelo, B. Schrauwen, and D. Stroobandt, "Event detection and localization for small mobile robots using reservoir computing," *Neural Networks*, vol. 21, no. 6, pp. 862–871, 2008.

[36] C. Hartland, N. Bredeche, and M. Sebag, "Memory-enhanced evolutionary robotics: the echo state network approach," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 2788–2795, IEEE, New York, NY, USA, May 2009.

[37] G. Holzmann and H. Hauser, "Echo state networks with filter neurons and a delay&sum readout," *Neural Networks*, vol. 23, no. 2, pp. 244–256, 2010.

[38] Y. Itoh and M. Adachi, "Chaotic time series prediction by combining echo-state networks and radial basis function networks," in *Proceedings of the IEEE 20th International Workshop on Machine Learning for Signal Processing (MLSP '10)*, pp. 238–243, New York, NY, USA, September 2010.

[39] H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," *Science*, vol. 304, no. 5667, pp. 78–80, 2004.

[40] D. Jing, G. K. Venayagamoorthy, and R. G. Harley, "Harmonic identification using an echo state network for adaptive control of an active filter in an electric ship," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '09)*, pp. 634–640, IEEE, Atlanta, Ga, USA, June 2009.

[41] X. Lin, Z. Yang, and Y. Song, "Short-term stock price prediction based on echo state networks," *Expert Systems with Applications*, vol. 36, no. 3, pp. 7313–7317, 2009.

[42] M. C. Ozturk and J. C. Principe, "An associative memory readout for ESNs with applications to dynamical pattern recognition," *Neural Networks*, vol. 20, no. 3, pp. 377–390, 2007.

[43] L. Qin and B. Lei, "Distributed multiagent for NAO robot joint position control based on echo state network," *Mathematical Problems in Engineering*. In press.

[44] M. H. Tong, A. D. Bickett, E. M. Christiansen, and G. W. Cottrell, "Learning grammatical structure with echo state networks," *Neural Networks*, vol. 20, no. 3, pp. 424–432, 2007.

[45] Y. Xue, L. Yang, and S. Haykin, "Decoupled echo state networks with lateral inhibition," *Neural Networks*, vol. 20, no. 3, pp. 365–376, 2007.

[46] D. Arbuckle and A. A. G. Requicha, "Active self-assembly," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '04)*, pp. 896–901, May 2004.

[47] A. Grushin and J. A. Reggia, "Stigmergic self-assembly of prespecified artificial structures in a constrained and continuous environment," *Integrated Computer-Aided Engineering*, vol. 13, no. 4, pp. 289–312, 2006.

[48] A. Grushin and J. A. Reggia, "Automated design of distributed control rules for the self-assembly of pre-specified artificial structures," *Robotics and Autonomous Systems*, vol. 56, no. 4, pp. 334–359, 2008.

[49] E. Klavins, "Programmable self-assembly," *IEEE Control Systems Magazine*, vol. 27, no. 4, pp. 43–56, 2007.

[50] C. E. Martin and J. A. Reggia, "Self-assembly of neural networks viewed as swarm intelligence," *Swarm Intelligence*, vol. 4, no. 1, pp. 1–36, 2010.

[51] J. Werfel and R. Nagpal, "Extended stigmergy in collective construction," *IEEE Intelligent Systems*, vol. 21, no. 2, pp. 20–28, 2006.

[52] J. Bishop, S. Burden, E. Klavins et al., "Programmable parts: a demonstration of the grammatical approach to selforganization," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS '05)*, pp. 3684–3691, IEEE, New York, NY, USA, 2005.

[53] R. Groß, M. Bonani, F. Mondada, and M. Dorigo, "Autonomous self-assembly in swarm-bots," *IEEE Transactions on Robotics*, vol. 22, no. 6, pp. 1115–1130, 2006.

[54] E. Klavins, R. Ghrist, and D. Lipsky, "Graph grammars for self assembling robotic systems," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '04)*, pp. 5293–5300, IEEE, New Orleans, La, USA, April–May 2004.

[55] J. Nembrini, N. Reeves, E. Poncet, A. Martinoli, and A. Winfield, "Mascarillons: Flying swarm intelligence for architectural research," in *Proceedings of the IEEE swarm intelligence symposium (SIS' 05)*, pp. 233–240, IEEE, June 2005.

[56] M. Rubenstein, A. Cornejo, and R. Nagpal, "Programmable self-assembly in a thousand-robot swarm," *Science*, vol. 345, no. 6198, pp. 795–799, 2014.

[57] P. White, V. Zykov, J. Bongard, and H. Lipson, "Three dimensional stochastic reconfiguration of modular robots," in *Proceedings of Robotics: Science and Systems*, pp. 161–168, MIT Press, Cambridge, Mass, USA, 2005.

[58] D. H. Gracias, J. Tien, T. L. Breen, C. Hsu, and G. M. Whitesides, "Forming electrical networks in three dimensions by self-assembly," *Science*, vol. 289, no. 5482, pp. 1170–1172, 2000.

[59] D. A. Elizondo, R. Birkenhead, M. Góngora, E. Taillard, and P. Luyima, "Analysis and test of efficient methods for building recursive deterministic perceptron neural networks," *Neural Networks*, vol. 20, no. 10, pp. 1095–1108, 2007.

[60] S. Fahlman and C. Lebiere, "The cascade-correlation learning architecture," in *Advances in Neural Information Processing Systems II*, S. D. Touretzky, Ed., pp. 524–532, Morgan Kaufmann, San Francisco, Calif, USA, 1990.

[61] M. Frean, "The upstart algorithm: a method for constructing and training feedforward neural networks," *Neural Computation*, vol. 2, no. 2, pp. 198–209, 1990.

[62] B. Hassibi, D. G. Stork, and G. J. Wolff, "Optimal brain surgeon and general network pruning," in *Proceedings of the IEEE International Conference on Neural Networks (ICNN '93)*, vol. 1, pp. 293–299, IEEE, April 1993.

[63] M. Marchand, M. Golea, and P. Ruján, "A convergence theorem for sequential learning in two-layer perceptrons," *Europhysics Letters*, vol. 11, no. 6, pp. 487–492, 1990.

[64] J. C. Astor and C. Adami, "A developmental model for the evolution of artificial neural networks," *Artificial Life*, vol. 6, no. 3, pp. 189–218, 2000.

[65] A. Cangelosi, D. Parisi, and S. Nolfi, "Cell division and migration in a 'genotype' for neural networks," *Network: Computation in Neural Systems*, vol. 5, no. 4, pp. 497–515, 1994.

[66] J. Chval, "Evolving artificial neural networks by means of evolutionary algorithms with L-systems based encoding," Research Report, Czech Technical University, Prague, Czech Republic, 2002.

[67] P. Eggenberger, "Creation of neural networks based on developmental and evolutionary principles," in *Proceedings of the International Conference on Artificial Neural Networks (ICANN '97)*, W. Gerstner, A. Germond, M. Hasler, and J. Nicoud, Eds., pp. 337–342, Springer, Lausanne, Switzerland, October 1997.

[68] F. Gruau, "Genetic synthesis of modular neural networks," in *Proceedings of the 5th International Conference on Genetic Algorithms (ICGA '93)*, S. Forest, Ed., pp. 318–325, Morgan Kaufmann, San Francisco, Calif, USA, 1993.

[69] J.-Y. Jung and J. A. Reggia, "Evolutionary design of neural network architectures using a descriptive encoding language," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 676–688, 2006.

[70] H. Kitano, "Designing neural networks using genetic algorithms with graph generation system," *Complex Systems*, vol. 4, pp. 461–476, 1990.

[71] Y. Chauvin and D. Rumelhart, Eds., *Backpropagation: Theory, Architectures, and Applications*, Lawrence Erlbaum Associates, Hillsdale, NJ, USA, 1995.

[72] B. A. Whitehead and T. D. Choate, "Cooperative-competitive genetic evolution of radial basis function centers and widths for time series prediction," *IEEE Transactions on Neural Networks*, vol. 7, no. 4, pp. 869–880, 1996.

[73] L. Zhao and Y. Yang, "PSO-based single multiplicative neuron model for time series prediction," *Expert Systems with Applications*, vol. 36, no. 2, pp. 2805–2812, 2009.

[74] F. Gruau, D. Whitley, and L. Pyeatt, "A comparison between cellular encoding and direct encoding for genetic neural networks," in *Proceedings of the 1st Annual Conference on Genetic Programming (GECCO '96)*, pp. 81–89, MIT Press, Cambridge, Mass, USA, 1996.

[75] F. Jiang, H. Berry, and M. Schoenauer, "Supervised and evolutionary learning of echo state networks," in *Parallel Problem Solving from Nature—PPSN X*, vol. 5199 of *Lecture Notes in Computer Science*, pp. 215–224, Springer, Berlin, Germany, 2008.

[76] A. P. Wieland, "Evolving neural network controllers for unstable systems," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '91)*, vol. 2, pp. 667–673, IEEE, New York, NY, USA, July 1991.

[77] D. Prokhorov, "Echo state networks: appeal and challenges," in *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN '05)*, vol. 3, pp. 1463–1466, IEEE, New York, NY, USA, July-August 2005.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 374873, 12 pages
http://dx.doi.org/10.1155/2015/374873

Research Article

Cuckoo Search Algorithm Based on Repeat-Cycle Asymptotic Self-Learning and Self-Evolving Disturbance for Function Optimization

Jie-sheng Wang,[1,2] Shu-xia Li,[1] and Jiang-di Song[1]

[1] *School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan 114044, China*
[2] *National Financial Security and System Equipment Engineering Research Center, University of Science and Technology Liaoning, Anshan 114044, China*

Correspondence should be addressed to Jie-sheng Wang; wang_jiesheng@126.com

Received 29 October 2014; Accepted 19 January 2015

Academic Editor: Rahib H. Abiyev

In order to improve convergence velocity and optimization accuracy of the cuckoo search (CS) algorithm for solving the function optimization problems, a new improved cuckoo search algorithm based on the repeat-cycle asymptotic self-learning and self-evolving disturbance (RC-SSCS) is proposed. A disturbance operation is added into the algorithm by constructing a disturbance factor to make a more careful and thorough search near the bird's nests location. In order to select a reasonable repeat-cycled disturbance number, a further study on the choice of disturbance times is made. Finally, six typical test functions are adopted to carry out simulation experiments, meanwhile, compare algorithms of this paper with two typical swarm intelligence algorithms particle swarm optimization (PSO) algorithm and artificial bee colony (ABC) algorithm. The results show that the improved cuckoo search algorithm has better convergence velocity and optimization accuracy.

1. Introduction

Cuckoo search (CS) algorithm, a new biological heuristic algorithm, is put forward by Yang and Deb in 2009. It simulates the cuckoo's seeking nest and spawning behavior and introduces Levy flight mechanism into it, which is able to quickly and efficiently find the optimal solution [1, 2]. Studies have proved that CS algorithm is better than other swarm intelligence algorithms in convergence rate and optimization accuracy, such as genetic algorithm (GA), particle swarm optimization (PSO) algorithm, and artificial bee colony (ABC) algorithm [3]. Because this algorithm has the characteristics of fewer parameters and being simple and easy to implement, now it has been successfully applied in a variety of engineering optimization problems. So the CS algorithm has a very high potential research value [4, 5].

CS algorithm is a new type of bionic algorithm. Many scholars carry out many researches in CS algorithm and put forward the corresponding improvement strategies. In literature [6], it gains insight into search mechanisms of CS algorithm and analyzes why it is efficient; meanwhile, it discusses the essence of algorithms and its link to self-organizing systems [6]. In literature [7], in order to increase CS efficiency, it exploits several parameters of the CS algorithm involving the Levy distribution factor beta (β) and the probability factor (P) and by seeking optimum values of these parameters efficiency of CS algorithm are improved [7]. In literature [8], it studied the algorithmic structure and behavior of CS and Levy distribution in detail, and then by comparing with widely used optimization algorithms (i.e., DE and GA) statistical results verified that CS has a more superior problem-solving ability [8]. For the purpose of enhancing the search ability of the cuckoo search (CS) algorithm, an improved robust approach, called harmony search (HS), is put forward, in which method a mutation operator is added to the process of the cuckoos updating to speed up convergence [9]. In literature [10], in order to solve combinatorial problems, it extends and improves CS by reconstructing its population and introducing a new category of cuckoos [10]. A cuckoo optimal algorithm based

on the exchange operator and the chaotic disturbance is proposed, which introduces the exchange operator theory of the particle swarm optimization algorithm to improve convergence rate and optimization accuracy [11]. A cooperative coevolutionary cuckoo search algorithm is put forward by applying the framework of cooperative coevolutionary, which divides the solution vectors of population into several subvectors and constructs the corresponding subswarms [12]. A novel cuckoo search optimization algorithm based on Gauss distribution is proposed by adding a Gauss distribution to CS algorithm to improve convergence rate [13]. A new self-adaptive cuckoo search algorithm is proposed by using a self-adaptive parameter control strategy to adjust the step size of CS to enhance its search ability [14]. Because the CS algorithm is highly random search according to Levy flight mechanism and shows a strong leaping, it is easy to jump from one region to another region, which makes the search that around each bird's nest location not careful and thorough and cannot make full use of information nearby the bird's nest locations. So the CS algorithm has characteristics of weak local searching ability, slow convergence rate, and low optimization accuracy.

In order to make up the defect of the algorithm on this aspect, a kind of improved cuckoo search algorithm based on the repeat-cycle asymptotic self-learning and self-evolving disturbance (RC-SSCS) is proposed. In order to obtain better disturbance effect, the learning and updating strategy of the worst frog in the shuffled frog leaping algorithm (SFLA) and part of differential evolution (DE) thought are introduced into the constructing of disturbance factor, thus, which makes every disturbance with the effect of nest's self-learning and self-evolving. Finally, the six typical test functions are chosen for simulation experiment and the simulation results proved that the improved cuckoo search algorithm has better convergence rate and optimization accuracy. The paper is organized as follows. In Section 2, the cuckoo search algorithm is introduced. The improved cuckoo search algorithm based on the repeat-cycle asymptotic self-learning and self-evolving disturbance is presented in Section 3. In Section 4, the simulation experiments and results analysis are introduced in details. Finally, the conclusion illustrates the last part.

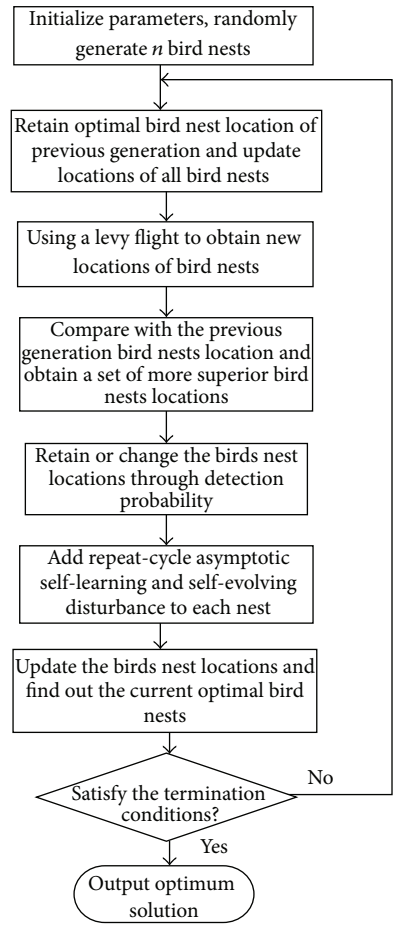


Figure 1: Flow chart of RC-SSCS algorithm.

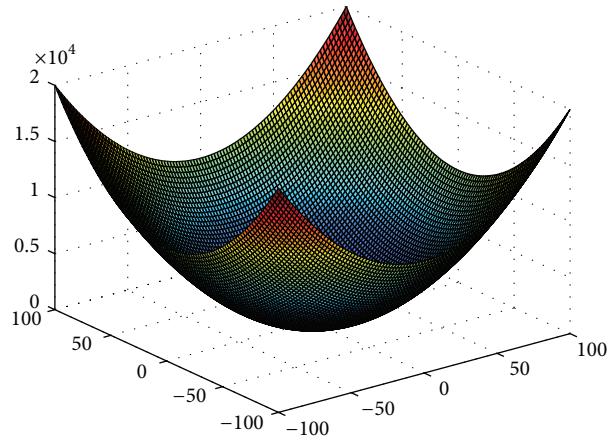


Figure 2: 3D surface figure of Sphere function.

2. Cuckoo Search Algorithm

Cuckoo search algorithm (CS) is developed by Xin-she Yang by observing the magical nature phenomenon and then giving an artificial processing, which is a new type of heuristic search algorithm [15, 16]. This algorithm is mainly based on two aspects: the cuckoo's parasitic reproduction mechanism and Levy flights search principle. In nature, cuckoos use a random manner or a quasirandom manner to seek bird's nest location [10]. Most of cuckoos lay their eggs in other bird nests and let the host raise their cubs instead of them. If a host found that the eggs are not its owns, it will either throw these alien eggs away from the nest or abandon its nest and build a new nest in other places. However, there are some cuckoos choosing nest that the color and shape of the host's egg are similar with their own to win the host's love, which can reduce the possibility of their eggs being abandoned and increase the reproduction rate of the cuckoos.

In general, each cuckoo can only lay one egg and each egg on behalf of one solution (cuckoo). The purpose is to make the new and potentially better solutions replace the not-so-good solutions (cuckoos). In order to study the cuckoo search algorithm better, the simplest method is adopted, that is to

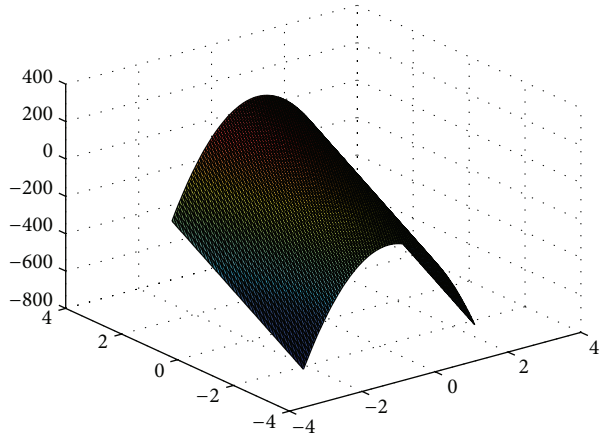


Figure 3: 3D surface figure of Rosenbrock function.

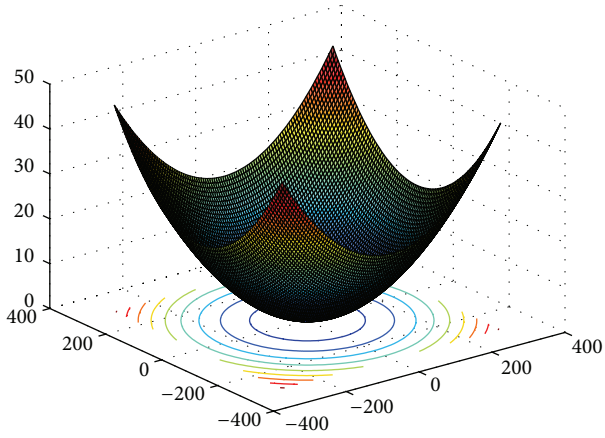


Figure 4: 3D surface figure of Griewank function.

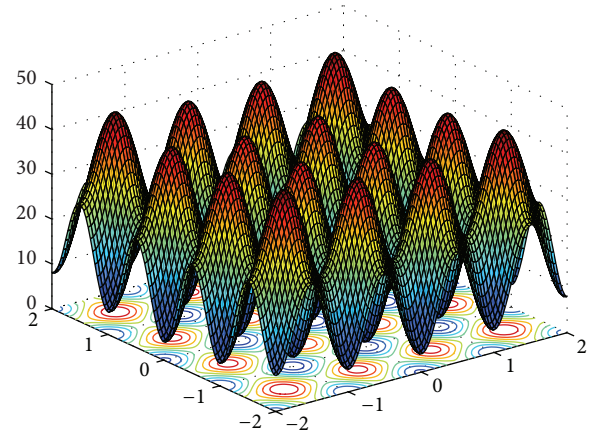


Figure 5: 3D surface figure of Rastrigrin function.

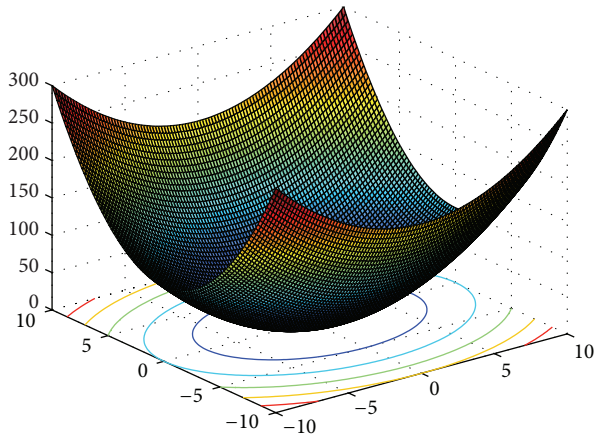


Figure 6: 3D surface figure of Sumsquares function.

say, only one egg is in each nest. In this case, an egg, a bird's nest, or a cuckoo is no different, which is to say, each nest corresponding to a cuckoo's egg. For simplicity in describing the cuckoo search algorithm, Yang and Deb use the following three idealized rules to construct the cuckoo algorithm [17].

(1) Each cuckoo only lays one egg at a time and randomly chooses bird's nest to hatch the egg.

(2) The best nest will carry over to the next generation.

(3) The number of available host nests is fixed, and the probability of a host discovering an alien egg is $P_a = [0, 1]$. In this case, the host bird may either throw the alien egg away or abandon its nest so as to build a new nest in a new location.

Under the above conditions, the specific steps of CS algorithm are described as follows.

(1) Initialization Setting. Randomly generate N bird's nest location $X_0 = (x_1^0, x_2^0, \ldots, x_N^0)$, and then take the N bird's nest positions into test functions for choosing experiments. Through testing, the best bird's nest location is chosen and carried over to the next generation.

(2) Searching of Bird's Nest Location. Equation (1) is used to realize the location update and search the bird's nest location of the next generation in order to gain a new set of bird's nest locations, which are taken into the test function again for testing experiments. After comparing with the last generation nest locations, the best bird's nest location is chosen and entered into the next step. Consider

$$x_i^{(t+1)} = x_i^t + \alpha \oplus \text{Levy}(\lambda), \quad i = 1, 2, \ldots, n. \tag{1}$$

(3) Selection of Bird's Nest Locations. The probability of a host discovering an alien egg $P_a = 0.25$ is compared with the random number that obeys uniform distribution $r \in [0, 1]$. If r, the value of x_i^{t+1} is randomly changed. Otherwise it does not need to be changed.

Then the changed bird's nest locations are calculated with the test functions, which are compared with the optimal position of previous generation and best bird's nest locations $X_t = (x_1^t, x_2^t, \ldots, x_N^t)$ is recorded. Finally, the optimal nest position pb_t^* is chosen.

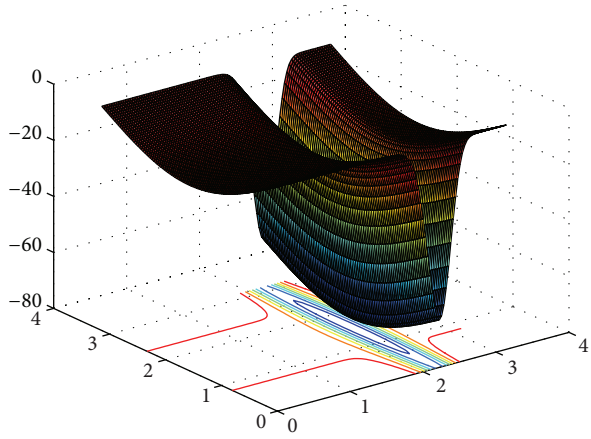


Figure 7: 3D surface figure of Michalewicz function.

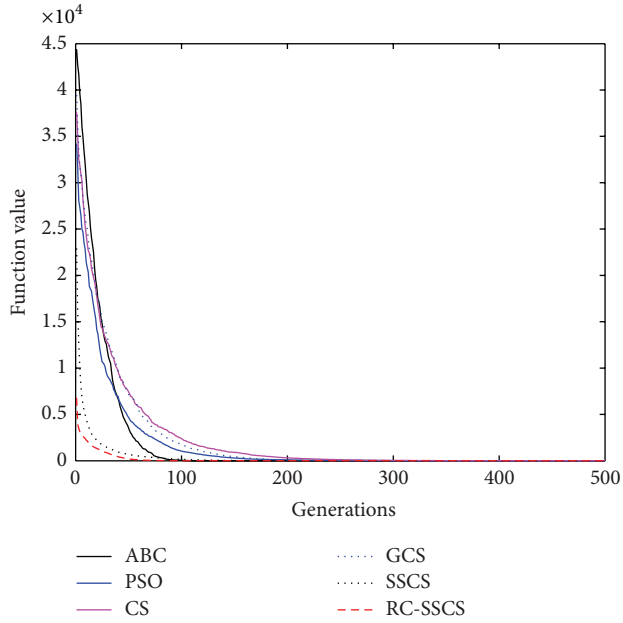


Figure 8: Convergence curves of Function f_1.

(4) Accuracy or Iterations Judgment. Calculate and judge whether $f(pb_t^*)$ achieves the object accuracy or the terminating conditions. If it meets the requirements, pb_t^* is the global optimal solution gb; if it is not be met, pb_t^* is kept to the next generation and return to step (2) and start the next loop iteration and update again.

According to these four steps described above of cuckoo search algorithm, the cuckoo algorithm not only uses the Levy flight (global search) search method, but also introduces the elite reserved strategy (local search), which makes the algorithm have both global search ability and local search ability. The purpose of step (3) in this algorithm is to increase the diversity of solution so that the algorithm is prevented from being caught into local optimum and achieving the global optimum.

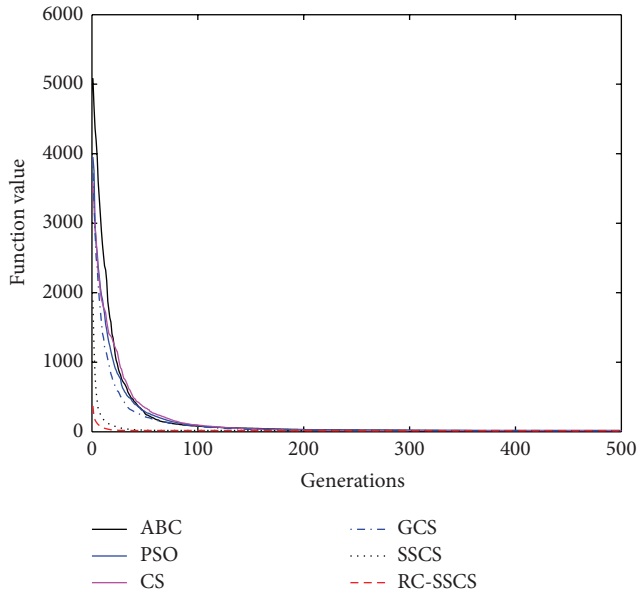


Figure 9: Convergence curves of Function f_2.

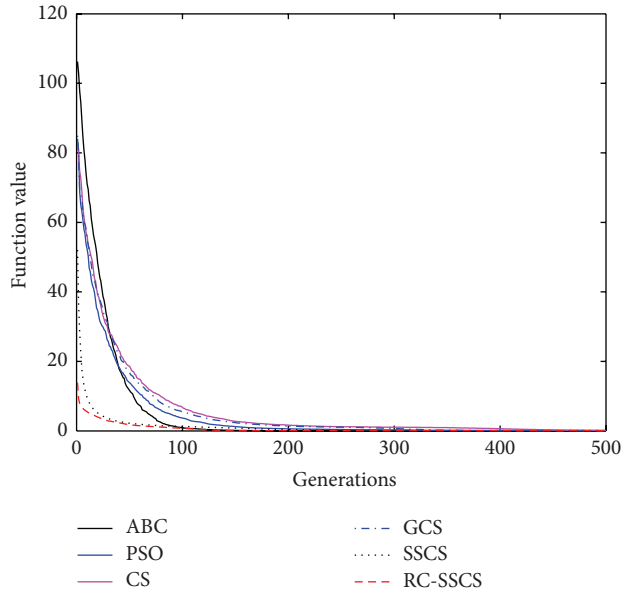


Figure 10: Convergence curves of Function f_3.

The search path of CS algorithm is different with other swarm algorithms. The CS algorithm uses Levy flight, which has character of strong randomness. Broadly speaking, Levy flight is a random walk, whose step size obeys Levy distribution, and the direction of travel is subject to uniform distribution. The step size vector of CS algorithm is determined by Mantegna rule of Levy distribution characteristics. In Mantegna rules, the step size s is designed as follows:

$$s = \frac{u}{|v|^{1/\beta}}, \tag{2}$$

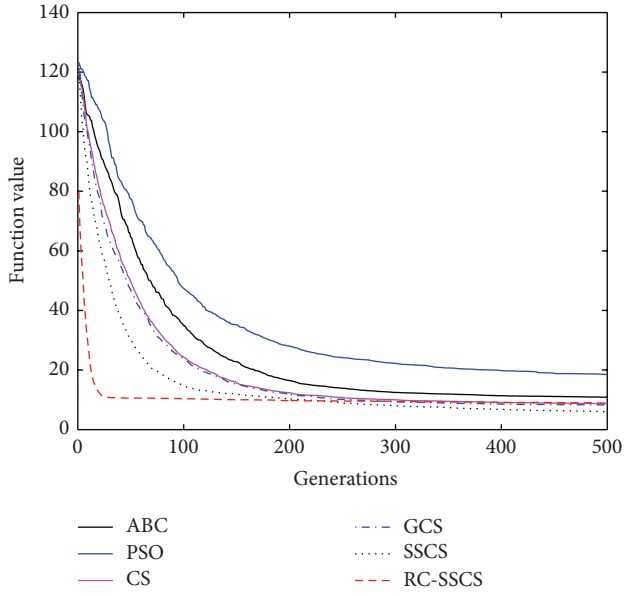


Figure 11: Convergence curves of Function f_4.

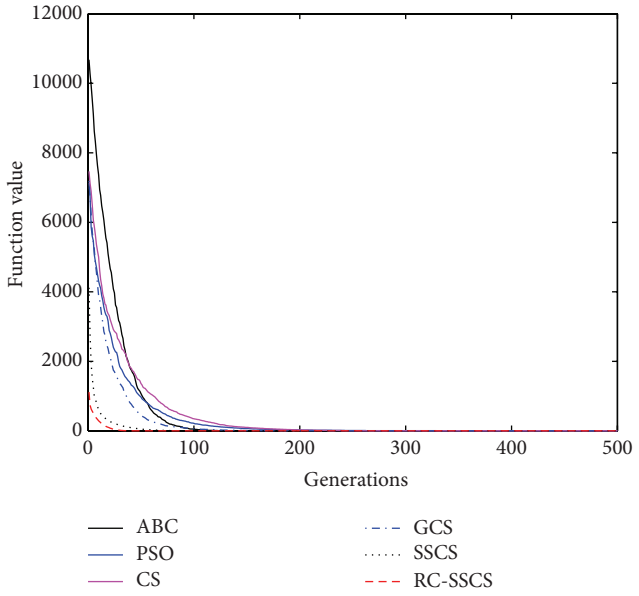


Figure 12: Convergence curves of Function f_5.

where u and v obey the normal distribution, that is to say

$$u \sim N\left(0, \sigma_u^2\right), \qquad v \sim N\left(0, \sigma_v^2\right), \tag{3}$$

where

$$\sigma_v = \left\{ \frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\Gamma\left[(1+\beta)/2\right]\beta 2^{(\beta-1)/2}} \right\}^{1/\beta}, \qquad \sigma_u = 1. \tag{4}$$

But here the chosen method of the direction obeys uniform distribution. The searching pattern of CS algorithm

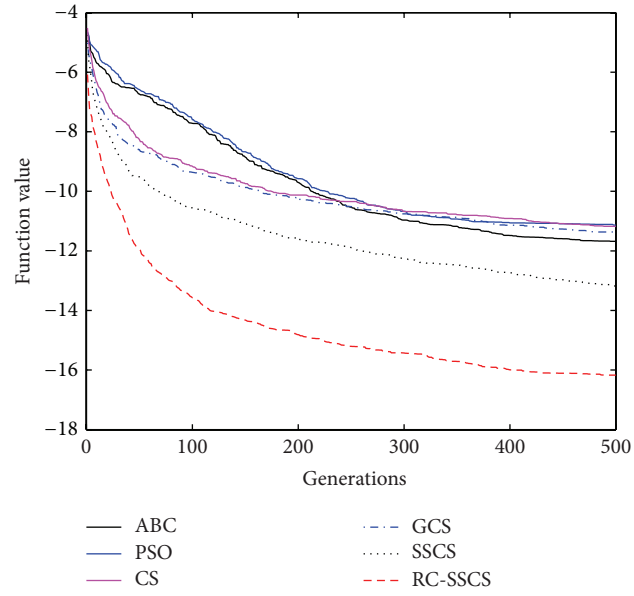


Figure 13: Convergence curves of Function f_6.

is Levy flight, for instance, the ith cuckoo of the tth generation generates the next generation solution x_i^{t+1}:

$$x_i^{t+1} = x_i^t + \alpha \oplus \text{Levy}(\lambda), \tag{5}$$

where $\oplus$ is a point to point multiplication and the step size of Levy(λ) obeys the Levy distribution, which can be expressed as follows:

$$\text{Levy} \sim u = t^{-\lambda}, \quad (1 < \lambda \leq 3). \tag{6}$$

Here, the Mantegna rules are used to calculate the step size. In (5), α is the step controlled quantity mainly used to control the direction and step size. In that its distribution is a power function, Levy distribution has infinite variance and its increment obeys heavy-tailed distribution [18, 19]. Levy flight is seemingly Brownian motion under the status of a long-distance flight, or it may be described that Levy flight consists of frequent short-jump and occasional long-jump. The long-jump can help the CS algorithm jump out of local optimum. Consider

$$\alpha = O\left(\frac{L}{10}\right), \tag{7}$$

where L is the search space size of optimization problems.

Thus, the generation of some new solutions is gradually through the Levy flight and rand walk around the optimal solution to obtain the optimal solution, which can speed up the local searching. On the contrary, a part of new solutions is far away from the current optimal solution, because they are randomly generated by deviating from remote locations. The main purpose of these solutions is to ensure that the system does not fall into the local optimal solution.

A large number of simulation experiments prove that when the bird's nest groups values $n = 15\sim40$ and the detection probability $P_a = 0.25$, it can solve many optimization

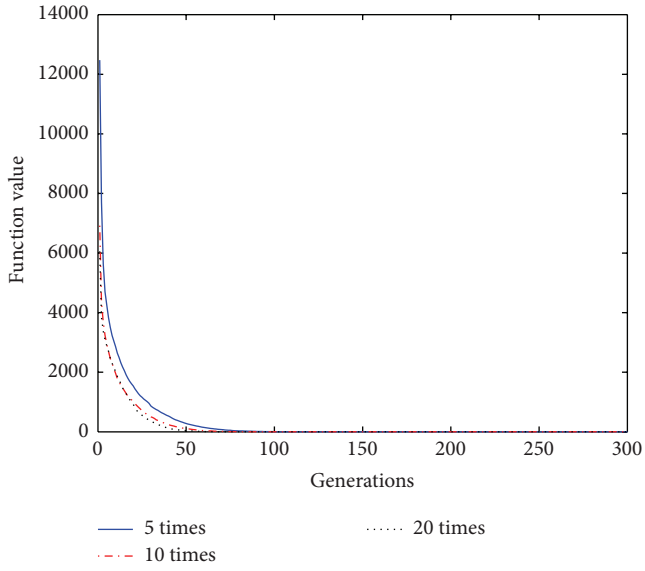


Figure 14: Convergence curves of Function f_1 under different disturbance number.

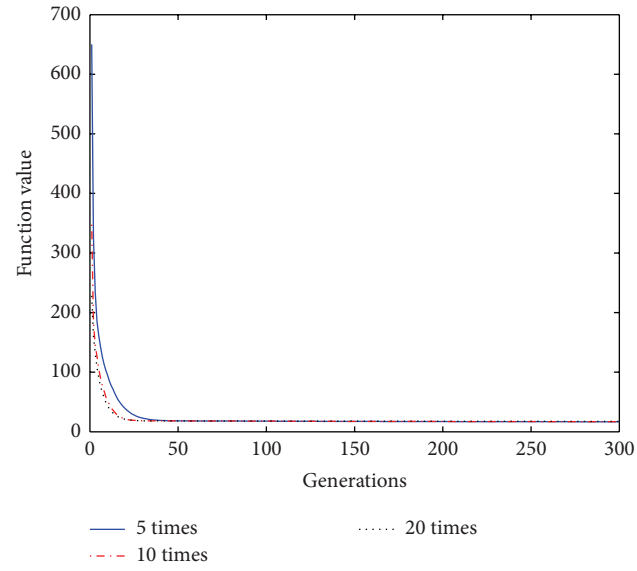


Figure 15: Convergence curves of Function f_2 under different disturbance number.

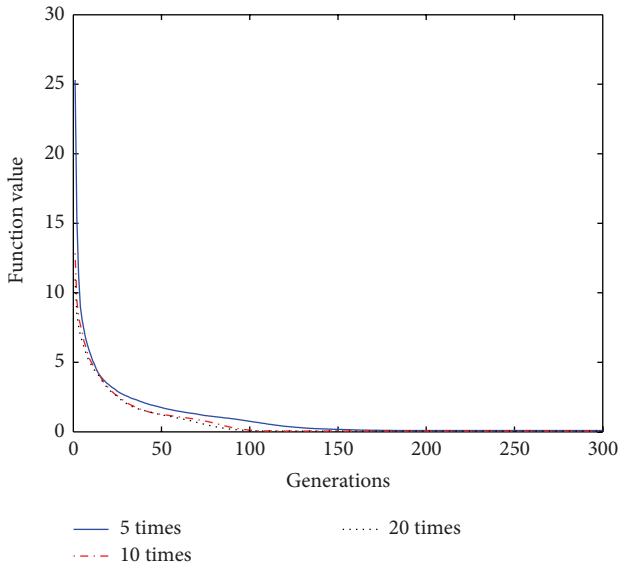


Figure 16: Convergence curves of Function f_3 under different disturbance number.

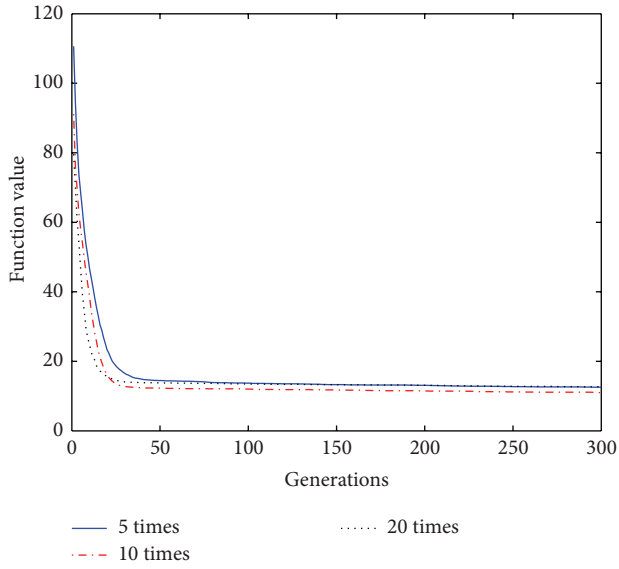


Figure 17: Convergence curves of Function f_4 under different disturbance number.

problems [16]. Once the population size n is fixed, the discovery probability P_a is an important parameter to balance the global search and the local search and control the elite selection. Therefore, the CS algorithm has characteristic of less parameters, excellent searching path and strong global optimization ability, and so forth [20].

3. The Improved Cuckoo Search Algorithm

For each time, the length and direction of cuckoos' searching path are highly randomly changed based on Levy flight mechanism, so they are easy to jump from one region to another, which is beneficial to the global search in the early stage of optimization and make the CS algorithm have strong global search ability [16, 21]. It is just because the CS algorithm shows a strong jumping in the search process that makes the local search around each bird's nest location no careful and no thorough. Therefore, the local optimization information near bird's nest location has not been fully utilized, which leads to that the local search ability is not strong, the optimization accuracy of the later period is not high, and the convergence speed is slow. In order to improve the convergence velocity and the optimization accuracy of the CS algorithm, a self-learning and self-evolving disturbance operation is added to

Table 1: Typical test functions.

Function name	Expression	Scope
Sphere	$f_1(\mathrm{x}) = \sum_{i=1}^{d} x_i^2$	$[-100, 100]$
Rosenbrock	$f_2(\mathrm{x}) = \sum_{i=1}^{d-1} \left(100\left(\mathrm{x}_{i+1} - x_i^2\right) + \left(\mathrm{x}_i - 1\right)^2\right)$	$[-2.08, 2.08]$
Griewank	$f_3(\mathrm{x}) = \frac{1}{4000}\left(\sum_{i=1}^{d}\left(x_i^2\right)\right) - \left(\prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right)\right) + 1$	$[-300, 300]$
Rastrigrin	$f_4(\mathrm{x}) = \sum_{i=1}^{d}\left(x_i^2 - 10\cos\left(2\pi \mathrm{x}_i\right) + 10\right)$	$[-1.25, 1.25]$
Sumsquares	$f_5(x) = \sum_{i=1}^{d} i x_i^2$	$[-10, 10]$
Michalewicz	$f_6(x) = -\sum_{i=1}^{d} sin(x_i)\left[sin\left(\frac{ix_i^2}{\pi}\right)\right]^{2m}, (m = 10)$	$[0, \pi]$

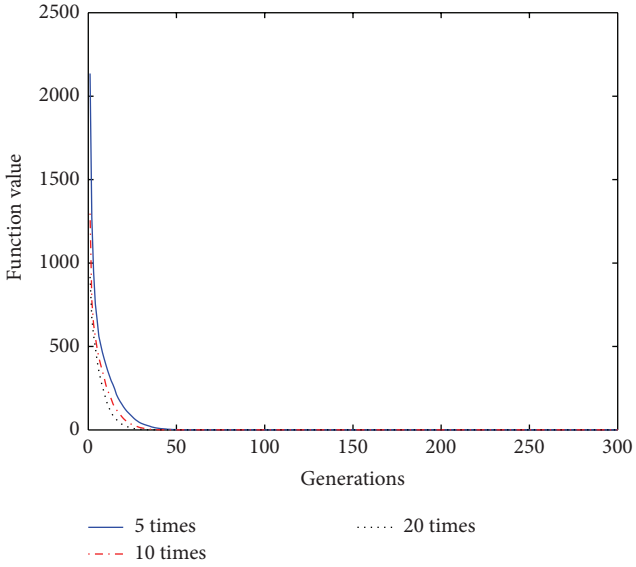


Figure 18: Convergence curves of Function f_5 under different disturbance number.

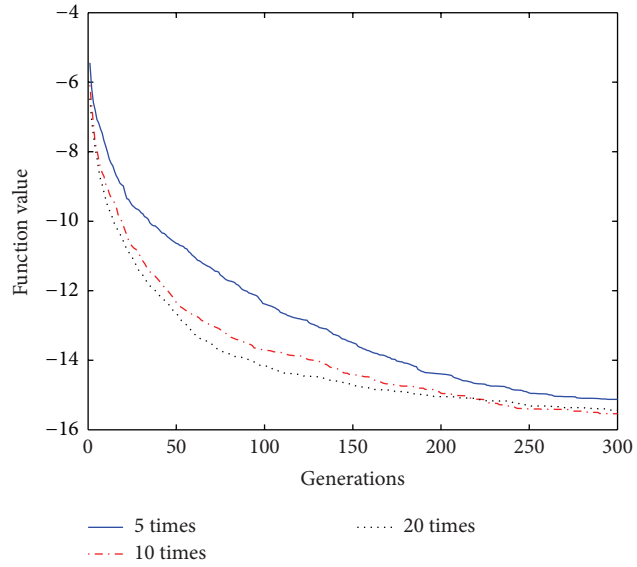


Figure 19: Convergence curves of Function f_6 under different disturbance number.

the algorithm, and a further study for the improvement of disturbance is also discussed.

3.1. Cuckoo Search Algorithm Based on Self-Learning and Self-Evolving Disturbance. In order to make the algorithm carry on more careful and thorough searches near bird's nest locations, after each iteration of CS algorithm a set of obtained preponderant bird's nest locations $X_i(t)$ instead of letting $X_i(t)$ directly into the next iteration, a disturbance operation is applied to it for making a further search on the neighborhood of $X_i(t)$.

Due to the general disturbance, such as Gauss perturbation and random perturbation, all having the great randomness and blindness, in order to obtain a better disturbance effect, the learning and updating strategy of the worst frog in the shuffled frog leaping algorithm (SFLA) and a part of differential evolution (DE) thought are introduced into the constructing of the disturbance factor. The introduction of learning and updating strategy of the worst frog can increase bird's self-learning ability of each bird's nest learning from the optimal nest [22]. That is to say it can increase the speed of other solutions approaching to the best solution and improve the algorithm's convergence rate. The introduction of the differential evolution thought can increase the diversity of bird's nest location, which makes every bird's nest have the evolution ability. The good learning and evolving ability are bound to cause high search ability.

Based on the above analysis, the disturbance factor is constituted of two parts: one part is the learning factor from SFLA learning and updating strategy of the worst frog; the other part is the evolution factor based on a part of differential evolution (DE) thoughts. Thus, the whole disturbance factor makes every disturbance with the effect of bird's self-learning

Table 2: Comparison of numerical testing results.

Function	Algorithm	Best	Worst	Mean
f_1	ABC	$3.1519e-004$	0.0704	$9.0266e-004$
	PSO	0.0048	7.0889	0.6284
	CS	0.0382	0.7001	0.2170
	GCS	$5.8627e-004$	0.0047	0.0016
	SSCS	$4.8773e-007$	$9.1317e-005$	$1.6997e-005$
	RC-SSCS	$3.6450e-012$	$3.0061e-005$	$1.7847e-008$
f_2	ABC	11.3082	18.1271	13.7651
	PSO	10.4223	34.5360	17.4946
	CS	11.1806	17.3151	15.2039
	GCS	8.5219	16.8620	14.2017
	SSCS	6.2653	14.5681	13.3126
	RC-SSCS	3.3432	14.2079	10.8277
f_3	ABC	$4.6297e-005$	0.6980	0.1859
	PSO	$5.8949e-005$	1.8793	0.3322
	CS	0.0189	0.3452	0.2010
	GCS	$2.7105e-004$	0.0077	0.1057
	SSCS	$2.2377e-009$	0.3909	0.0814
	RC-SSCS	$1.0616e-011$	0.1954	0.0454
f_4	ABC	5.2919	24.7673	10.3917
	PSO	6.0188	29.7667	18.5742
	CS	5.0454	11.4347	9.4766
	GCS	4.5346	9.4692	8.7840
	SSCS	1.2841	9.3480	6.0351
	RC-SSCS	0.6950	10.3400	5.8859
f_5	ABC	$2.5878e-005$	$8.4891e-004$	$3.3576e-004$
	PSO	$5.6717e-004$	4.2837	0.2335
	CS	$8.7353e-004$	0.0068	0.0030
	GCS	$1.5714e-004$	$9.2999e-004$	$4.2927e-004$
	SSCS	$9.3420e-007$	$9.5495e-004$	$9.8643e-005$
	RC-SSCS	$2.4677e-009$	$8.4555e-004$	$5.6853e-006$
f_6	ABC	−14.7812	−8.1498	−11.0840
	PSO	−15.1576	−7.8576	−10.8402
	CS	−12.7182	−10.1137	−11.3479
	GCS	−13.9297	−10.4565	−11.6951
	SSCS	−15.1779	−11.4004	−13.6241
	RC-SSCS	−18.0889	−13.1681	−16.7740

and self-evolving ability. Disturbance factor is structured as follows:

$$\varepsilon = c_1 u_1 \left(x_i - x_{\text{best}}\right) + c_2 u_2 \left(x_{r_1} - x_{r_2}\right), \tag{8}$$

where x_i is the ith disturbed bird's nest location, x_{best} is the current best location, r_1 and r_2 are random number from $(1, n)$, $r_1 \neq r_2$, n is the number of bird's nest population, x_{r_1} and x_{r_2} are the bird's nest locations corresponded to a random number r_1 and r_2, u_1, and u_2 obey the Gaussian distribution, c_1 is the learning scale, and c_2 is evolution scale.

For better controlling of the disturbance range, a controlled quantity of disturbance scope γ is introduced to control the search scope size around a bird's nest. After being disturbed, the bird's nest location is express as

$$x_i' = x_i + \gamma \oplus \varepsilon, \tag{9}$$

where x_i' is the bird's nest location after being disturbed, γ is the controlled quantity of the disturbance range, and $\oplus$ is the point to point multiplication.

According to (8) and (9), after searching and selection operation obtain a bird's nest location x_i, do not let this nest go into the next generation directly. Instead, add to it a disturbance that take x_i as the foundation and ε disturbance factor, within the distribute scope that is controlled by γ. Finally after disturbance get a new bird's nest location x_i', and let x_i' go into the next generation.

Table 3: Comparison of numerical testing results under different cycled disturbance number.

Function	Cycle number	Best	Worst	Mean
f_1	5	$1.0950e-006$	$4.5927e-004$	$8.4772e-005$
	10	$8.0914e-008$	$3.9593e-005$	$3.3471e-006$
	20	$8.1817e-007$	$1.1113e-004$	$2.4506e-005$
f_2	5	15.5346	19.4028	18.2092
	10	14.5140	19.4748	16.6741
	20	14.9356	19.4993	16.6707
f_3	5	$2.0991e-007$	1.2703	0.1498
	10	$1.0955e-008$	0.3909	0.0847
	20	$3.1836e-008$	1.3680	0.0847
f_4	5	5.9698	16.9143	12.4617
	10	1.9899	14.9247	11.2695
	20	1.9900	17.9210	11.7947
f_5	5	$1.5066e-007$	$1.3715e-004$	$5.1063e-005$
	10	$5.5967e-008$	0.0052	$3.6228e-005$
	20	$3.2597e-007$	0.0022	$2.9710e-004$
f_6	5	−17.3839	−11.9059	−15.1020
	10	−18.2436	−12.5260	−15.5739
	20	−18.0307	−13.0349	−15.5256

In different stages, the bird nest's learning ability, the evolving ability, and the disturbance scope can be controlled by adjusting the values of c_1, c_2, and γ. For example, when γ is a given value, if $c_1 = 0$ and c_2 is a certain constant, then within this scope of search, the bird's nest only has evolution ability, without learning ability; by the same token, if $c_2 = 0$ and c_1 is a certain constant, within the search scope, the algorithm only has learning ability and no evolving ability.

The disturbance range controller γ not only controls the search scope size, but also affects a bird nest's learning ability and evolution ability. When γ is given a big value, the coefficient before learning factor $x_i - x_{\text{best}}$ and evolving factor $x_{r_1} - x_{r_2}$ can be driven larger, so the learning and evolution ability become stronger. Under this condition, the search scope of the algorithm is larger, the search ability is stronger but the search fineness is lower. By the same token, when γ is given a small value, the learning and evolution ability are relatively weaker, the search scope is smaller, the search ability becomes weaker, but the search fineness is much higher.

3.2. Cuckoo Search Algorithm Based on Repeat-Cycle Asymptotic Self-Learning and Self-Evolving Disturbance. The ideal disturbance effect should be that in the early stage of disturbances, it has higher searching ability and in the later stage of disturbances has higher search accuracy. In this way, it can obtain a better bird's nest location fast and then carry on a more careful and thorough search around the better nest location. Based on the above analysis, a repeat-cycle asymptotic disturbance method is proposed. Adopting a dynamic adjustment γ makes the search scope gradually change from big to small after disturbance. In other words, based on the results of last disturbance, narrowing the disturbance scope, go on the next disturbance. The repeat-cycle asymptotic disturbance search is carried out in turn.

In order to obtain a better bird's nest position at a faster speed, at the beginning of disturbance γ is given a larger value. With the disturbance continuing, the bird's nests gradually get better and the adjustment of bird's nest locations is more and more subtle. Particularly in the condition that the response of fitness value is sensitive to parameter changing, it needs to use a very small amount of control γ to make the position have a fine-tuning near optimal value. The γ is adjusted according to the following:

$$\gamma = \gamma_{\min} + (\gamma_{\max} - \gamma_{\min}) \times \frac{N - n}{N}, \tag{10}$$

where $\gamma_{\min}$ is the minimum value of control amount of disturbance scope, $\gamma_{\max}$ is the maximum value, N is the total number of repeat-cycle disturbances, and n is the nth disturbance.

In the circulation disturbance, the controlled quantity of the disturbance range γ is controlled by disturbance number n. Make γ changes between $\gamma_{\max}$ and $\gamma_{\min}$. When $n = 1$, γ is maximum $\gamma = \gamma_{\max}$. With the disturbance number n increasing gradually γ decrease little by little. Finally when $n = N$, γ reaches minimum value $\gamma = \gamma_{\min}$.

In the early stage of the repeat-cycle disturbances, the bird's nest self-learning and self-evolving function plays a leading role, which can make it surely find a better bird's nest faster. With disturbance number increasing, the search scope gradually turns smaller and the fineness degree of search is gradually enhanced. In the later stage of the repeat-cycle disturbances, the fineness search plays a leading role.

3.3. Algorithm Procedure. The algorithm procedure of the improved cuckoo search algorithm (RC-SSCS) is shown in Figure 1. The specific steps of RC-SSCS algorithm are described as follows.

Step 1 (initialization). Randomly generate N bird's nest location $X^0 = (x_1^0, x_2^0, \ldots, x_N^0)$, select the best bird's nest location, and carry it over to the next generation.

Step 2 (searching operation). Use the location update (1) to search for the next generation bird's nest position, obtain a new set of bird's nests, and test them. Then compare the testing results with the bird's nest position of previous generation and obtain a set of better positions.

Step 3 (selection operation). Generate the random number $r \in [0, 1]$, which obeys the uniform distribution. Contrast it with the detection probability $P_a = 0.25$, if $r > P_a$, $x_i^{(t+1)}$ is changed randomly, otherwise $x_i^{(t+1)}$ is unchanged. Test the changed nest positions, compare them with locations of the last step, and choose the better nest locations.

Step 4 (repeat-cycle disturbance operation). Add a self-learning and self-evolving disturbance to each bird's nest location, test new bird's nest locations that have been disturbed, and then compare the test results with locations of the last disturbance results and choose the better bird's nest locations. After many times disturbance, obtain a set of the best bird's nest locations, and then choose a best location pb from the set.

Step 5 (judgment operation). Calculate the fitness value of $f(pb)$ and judge whether it achieves the termination condition. If it is satisfied, pb is the optimal solution, otherwise return to Step 2 and start the next iteration.

4. Simulation Results and Analysis

In order to verify the performances of the improved CS algorithm, six typical continuous test functions are chosen for carrying out the simulation research, meanwhile, which is compare simulation results with ABC, PSO, CS and GCS. These six test functions are shown in Table 1. Their 3D surface figures are shown in Figures 2–7.

Sphere function is a simple unimodal function. Rosenbrock function is an inseparable single mode function, and its global extreme value is in steep valleys. For the most search algorithms, it is difficult to acquire the right search direction within the canyon. Griewank function is a multimodal function with multiple local optimal points, and due to the correlation between variables, it will be very hard to obtain the global optimal solution. Rastrigrin function is a typical inseparable multimodal function, and in its searching domain, there are a large number of local minimum values, which leads to the fact that it is difficult to obtain the global optimum. Michalewicz function has $d!$ local extreme values.

The experimental parameters are set as follows. For particle swarm optimization (PSO) algorithm the particle number is $n = 30$; learning factors $c_1 = 2$, $c_2 = 2$; the inertia weight $w = 0.9$. For artificial bee colony (ABC) algorithm total number of colonies is $n = 20$, the number of following bees and leading bees is the same $n/2 = 10$. For cuckoo search (CS) algorithm and its 3 improved CS algorithm the total bird's nest population $n = 25$, the detection probability $P_a = 0.25$, the step length controlled parameter $\alpha = 0.01$. The cuckoo search algorithm based on Gauss disturbance (GCS) and the cuckoo search algorithm based on self-learning and self-evolving disturbance (SSCS) adopt the same disturbance scope control quantity $\gamma = 0.75$. For the cuckoo search algorithm based on repeat-cycle asymptotic self-learning and self-evolving disturbance (RC-SSCS), the scope of γ is [0.25, 1.5], the scale of learning and evolution is set $c_1 = c_2 = 0.75$. The number of the circulation disturbance $N = 10$. For all algorithms, the dimension of these six test functions is all set $D = 20$. The number of iterations iter $= 500$. For each algorithm, its program run 30 times independently.

Evaluate the performances of algorithms through statistics of the best value, average value and worst value among 30 times running, and the convergence curves of each function. The convergence curves of six functions f_1–f_6 are shown in Figures 8–13. The numerical test results of each algorithm are shown in Table 2.

After 500 iterations and 30 times independently running, it can be seen from the convergence curves and numerical statistics results of six functions that the convergence rate and the optimization accuracy of RC-SSCS algorithm is the best. And the convergence rate and optimization accuracy of the two algorithms SSCS and RC-SSCS proposed in this paper are obviously better than the original CS algorithm, GCS algorithm, ABC algorithm, and PSO algorithm.

Seen from six convergence curves, the convergence rate of the six functions is all obviously improved. The convergence curves of function f_2 show that the optimization accuracy achieved by RC-SSCS algorithm after 20 times iteration is equal to that achieved by GCS algorithm and CS algorithm after 200 iterations. The function f_3 convergence curve shows that the optimization accuracy by RC-SSCS algorithm after 100 iterations reached is equal to the accuracy by original CS algorithm after 400 iterations reached. From the convergence curve of function f_4, It can be seen that the optimization accuracy achieved by RC-SSCS algorithm after 25 iterations is equal to that by SSCS algorithm after 175 iterations achieved and by CS algorithm after 300 iterations achieved. The convergence rate of function f_5 and f_6 is also changed obviously. The results show that the improved algorithm makes the convergence rate be greatly improved.

Seen from the numerical results of Table 2, the improved algorithm makes the optimization accuracy of six typical functions be improved. The SSCS algorithm and RC-SSCS algorithm relative to CS algorithm, respectively, make the best value of single-mode function f_1 increased by 5 and 10 orders of magnitude, and the average value, respectively, increased by 4 and 7 orders of magnitude. For multimode function f_3 with multiple local optimal points, compared with the original CS algorithm, SSCS, and RC-SSCS algorithm, respectively, make its best value and average value improved by 5 and 9 orders of magnitude. Optimization accuracy of other functions has been improved. It shows that the improved algorithm can make the optimization accuracy be improved.

Based on the above analysis of the improved algorithm, it is known that, within a certain range of disturbance, the

number of the repeat-cycle disturbance affects the balance between the search capability and the optimization accuracy in disturbance process. An appropriate disturbance time can make a good balance between search ability and search accuracy and play a best optimization effect. In process of disturbance search, if the algorithm only has strong search ability but no high search precision, it will not get an ideal search effect, Similarly, if the precision is very high, but search ability is weak, it will be also no ideal search effect. In order to select a reasonable disturbance time, in this paper we carry out the further studies on the relationship between the repeat-cycle disturbance times and the convergence rate and optimization precision. The same parameter settings are chosen as described above, except the number of iterations set iter = 300. The cycle disturbances to each function are carried out 5 times, 10 times, and 20 times, respectively. The results of the six functions under different disturbance are shown in Figures 14–19 and Table 3.

It can be seen from the convergence curves in Figures 14–19 that with the increase of cycle's times, the convergence rate of six functions will be gradually improved. But by looking carefully at each function convergence curve, it can be discovered that the changing of convergence rate when number of loops increases from 5 to 10 is bigger than that when loops number increases from 10 to 20. That is to say, the changing of convergence produced by increasing 5 times loops in front of repeat-cycle disturbance is bigger than that by increasing 10 times in later of it. It also can be seen from the numerical results in Table 3 that the optimization accuracy is the highest when cycle disturbance times is 10. The optimization accuracy of six functions is all improved when the disturbance times are increased from 5 to 10. However, when the disturbance times are increased from 10 to 20, the optimization accuracy all decreases instead of increasing.

In conclusion, the more disturbance times may not obtain the better results. Within the same disturbance scope, when the disturbance times reach a certain number, if the disturbance number increases again, the convergence speed of the algorithm does not have an obvious improvement. However, the optimization accuracy will be reduced. Integrally considering the convergence rate, the optimization accuracy and the optimizing time, it is better to choose the cycled disturbance times about 10.

5. Conclusion

In order to improve the convergence rate and optimization accuracy of the cuckoo search (CS) algorithm for solving function optimization problems, a kind of cuckoo search algorithm based on the repeat-cycle asymptotic self-learning and self-evolving disturbance (RC-SSCS) is proposed. Six typical test functions are chosen for simulation experiments. Simulation results show the effectiveness of the proposed improved cuckoo search algorithm in convergence rate and optimization accuracy.

Conflict of Interests

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

Acknowledgments

This work is partially supported by The Program for China Postdoctoral Science Foundation (Grant no. 20110491510), The Program for Liaoning Excellent Talents in University (Grant no. LR2014008), and The Project by Liaoning Provincial Natural Science Foundation of China (Grant no. 2014020177).

References

[1] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, IEEE Publications, December 2009.

[2] X.-S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 4, pp. 330–343, 2010.

[3] X.-S. Yang and S. Deb, "Multiobjective cuckoo search for design optimization," *Computers & Operations Research*, vol. 40, no. 6, pp. 1616–1624, 2013.

[4] X. S. Yang, "Cuckoo search and firefly algorithm: overview and analysis," in *Cuckoo Search and Firefly Algorithm*, vol. 516 of *Studies in Computational Intelligence*, pp. 1–26, Springer International Publishing, Cham, Switzerland, 2014.

[5] E. Valian, S. Mohanna, and S. Tavakoli, "Improved cuckoo search algorithm for feedforward neural network training," *International Journal of Artificial Intelligence & Applications*, vol. 2, no. 3, pp. 36–43, 2011.

[6] X.-S. Yang and S. Deb, "Cuckoo search: recent advances and applications," *Neural Computing and Applications*, vol. 24, no. 1, pp. 169–174, 2014.

[7] A. Mallick, S. Roy, S. S. Chaudhuri, and S. Roy, "Study of parametric optimization of the Cuckoo Search algorithm," in *Proceedings of the International Conference on Control, Instrumentation, Energy and Communication (CIEC '14)*, pp. 767–772, Calcutta, India, January-February 2014.

[8] P. Civicioglu and E. Besdok, "Comparative analysis of the cuckoo search algorithm," in *Cuckoo Search and Firefly Algorithm*, vol. 516 of *Studies in Computational Intelligence*, pp. 85–113, Springer International Publishing, 2014.

[9] G.-G. Wang, A. H. Gandomi, X. Zhao, and H. C. E. Chu, "Hybridizing harmony search algorithm with cuckoo search for global numerical optimization," *Soft Computing*, 2014.

[10] A. Ouaarab, B. Ahiod, and X.-S. Yang, "Discrete cuckoo search algorithm for the travelling salesman problem," *Neural Computing and Applications*, vol. 24, no. 7-8, pp. 1659–1669, 2014.

[11] C.-W. Qu, Y.-M. Fu, and X.-L. Hang, "Cuckoo optimal algorithm based on exchange operator and chaotic disturbance," *Journal of Chinese Computer Systems*, vol. 2, no. 27, pp. 384–387, 2014.

[12] X.-X. Hu and Y.-L. Yin, "Cooperative co-evolutionary cuckoo search algorithm for continuous function optimization problems," *Pattern Recognition and Artificial Intelligence*, vol. 26, no. 11, pp. 1041–1049, 2013.

[13] H. Zheng and Y. Zhou, "A novel cuckoo search optimization algorithm based on Gauss distribution," *Journal of Computational Information Systems*, vol. 8, no. 10, pp. 4193–4200, 2012.

[14] W.-Y. Qian, H.-C. Hou, and S.-Y. Jiang, "New self-adaptive cuckoo search algorithm," *Computer Science*, vol. 41, no. 7, pp. 279–282, 2014.

[15] A. R. Yildiz, "Cuckoo search algorithm for the selection of optimal machining parameters in milling operations," *The International Journal of Advanced Manufacturing Technology*, vol. 64, no. 1–4, pp. 55–61, 2013.

[16] A. K. Bhandari, V. K. Singh, A. Kumar, and G. K. Singh, "Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy," *Expert Systems with Applications*, vol. 41, no. 7, pp. 3538–3560, 2014.

[17] J. Ahmed and Z. Salam, "A Maximum Power Point Tracking (MPPT) for PV system using Cuckoo Search with partial shading capability," *Applied Energy*, vol. 119, pp. 118–130, 2014.

[18] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems," *Engineering with Computers*, vol. 29, no. 1, pp. 17–35, 2013.

[19] A. Kaveh and T. Bakhshpoori, "Optimum design of steel frames using Cuckoo Search algorithm with Lévy flights," *The Structural Design of Tall and Special Buildings*, vol. 22, no. 13, pp. 1023–1036, 2013.

[20] P. Civicioglu and E. Besdok, "A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms," *Artificial Intelligence Review*, vol. 39, no. 4, pp. 315–346, 2013.

[21] R. R. Bulatović, S. R. Dordević, and V. S. Dordević, "Cuckoo Search algorithm: a metaheuristic approach to solving the problem of optimum synthesis of a six-bar double dwell linkage," *Mechanism and Machine Theory*, vol. 61, pp. 1–13, 2013.

[22] J. Luo and M.-R. Chen, "Improved shuffled Frog Leaping algorithm and its multi-phase model for multi-depot vehicle routing problem," *Expert Systems with Applications*, vol. 41, no. 5, pp. 2535–2545, 2014.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 364512, 11 pages
http://dx.doi.org/10.1155/2015/364512

Research Article

Expected Utility Based Decision Making under Z-Information and Its Application

Rashad R. Aliev,[1] Derar Atallah Talal Mraiziq,[1] and Oleg H. Huseynov[2]

[1]*Department of Mathematics, Eastern Mediterranean University, Famagusta, Northern Cyprus, Mersin 10, Turkey*
[2]*Department of Computer-Aided Control Systems, Azerbaijan State Oil Academy, 20 Azadlig Avenue, 1010 Baku, Azerbaijan*

Correspondence should be addressed to Rashad R. Aliev; rashad.aliyev@emu.edu.tr

Received 19 September 2014; Accepted 25 December 2014

Academic Editor: Rahib H. Abiyev

Real-world decision relevant information is often partially reliable. The reasons are partial reliability of the source of information, misperceptions, psychological biases, incompetence, and so forth. Z-numbers based formalization of information (Z-information) represents a natural language (NL) based value of a variable of interest in line with the related NL based reliability. What is important is that Z-information not only is the most general representation of real-world imperfect information but also has the highest descriptive power from human perception point of view as compared to fuzzy number. In this study, we present an approach to decision making under Z-information based on direct computation over Z-numbers. This approach utilizes expected utility paradigm and is applied to a benchmark decision problem in the field of economics.

1. Introduction

Decision making is one of the attractive research areas in the last decades. The complexity and uncertainty are persistent phenomenon in the real world, and the fuzzy set [1–3] is widely used in decision making process [3, 4]. Much of decision based information is uncertain. Human has a high capability of making logical decisions based on uncertain, incomplete, and/or inaccurate information [5].

Z-number is a sufficient formalization of real-world information that should roughly be considered in light of its reliability. The critical issue is that the reliability of information is not considered properly. Zadeh has proposed a new notion Z-number which is more appropriate to describe the uncertainty. Z-number takes both restraint and reliability. In comparison with the classical fuzzy number, Z-number has more ability to describe the real information of human [6].

Z-numbers were firstly presented by Zadeh in 2011 [5], and afterwards the researchers started to discuss Z-numbers in decision making under uncertainty and in many other fields. One of the main goals of Z-number is to produce fuzzy numbers with degree of self-confidence in order to know the real information. By using the Z-number the knowledge of human can be represented in a better way [7].

The computations with Z-numbers can be viewed as a generalization of computations with numbers, intervals, fuzzy numbers, and random numbers. As specified, the levels of generality can be separated as follows: computation with numbers (ground level zero); computation with intervals (level one); computation with fuzzy numbers (level two) [2]; computation with random numbers (level two); and computation with Z-numbers (level three). The capability of building realistic models of real-world systems is increased by the increase of the generality level, especially in the realms of economics, risk assessment, decision analysis, planning, and analysis of causality [5].

In [4] the authors suggest an approach to use Z-numbers for solving multicriteria decision making problem. For computation over Z-numbers some operations are suggested that are based on Zadeh's extension principle [5]. Z-numbers are also used for the purpose of reasoning [8]. In [6] proposed approach is intended to use Z-numbers for the expected utility application to solve decision making problems. An approach to use Z-numbers for answering questions and decisions making is considered in [9]. Z-numbers converted into classical fuzzy numbers are suggested in [4, 9]. In [7], Z-numbers are converted into classical fuzzy numbers and the fuzzy numbers are converted into crisp numbers. In [10]

the theoretical approach for computing arithmetic operations over discrete Z-numbers is proposed.

In [11] authors suggest general and computationally effective theoretic approach to computations with discrete Z-numbers. The authors provide strong motivation of the use of discrete Z-numbers as an alternative to the continuous counterparts. In particular, the motivation is based on the fact that NL based information has a discrete framework. The suggested arithmetic of Z-numbers includes basic arithmetic operations and important algebraic operations over Z-numbers. The proposed approach allows dealing with Z-information directly.

This paper focuses on investigating an approach for decision making which generalizes the expected utility approach of Z-information. This approach is based on direct computation over Z-numbers without converting them to fuzzy numbers and differed from the existing works used for decision making problems. The direct computation of Z-numbers without conversion eliminates the loss of information. In this research we recommend an approach based on expected utility to solve the decision making problems with Z-information. This approach is based on computation over Z-numbers according to operations suggested in [5, 10]. At the end, we provide a numerical example of the proposed approach to solve a benchmark problem.

This paper is organized as follows. The preliminaries for Z-numbers are reviewed in Section 2. Section 3 describes the numerical computations with discrete Z-numbers. Section 4 is devoted to statement and solution of a considered decision problem with Z-information. Section 5 consists of application, and the conclusions are revealed in Section 6.

2. Preliminaries

Definition 1 (a discrete fuzzy number [12–14]). A fuzzy subset A of the real line $\mathscr{R}$ with membership function $\mu_A : \mathscr{R} \to [0, 1]$ is a discrete fuzzy number if its support is finite; that is, there exist $x_1, \ldots, x_n \in \mathscr{R}$ with $x_1 < x_2 < \cdots < x_n$, such that $\text{supp}(A) = \{x_1, \ldots, x_n\}$ and there exist natural numbers s, t with $1 \le s \le t \le n$ satisfying the following conditions:

(1) $\mu_A(x_i) = 1$ for any natural number i with $s \le i \le t$;

(2) $\mu_A(x_i) \le \mu_A(x_j)$ for any natural numbers i, j with $1 \le i \le j \le s$;

(3) $\mu_A(x_i) \ge \mu_A(x_j)$ for any natural numbers i, j with $t \le i \le j \le n$.

Definition 2 (probability measure of a discrete fuzzy number [15]). Let A be a discrete fuzzy number. A probability measure of A denoted by $P(A)$ is defined as

$$\begin{aligned} P(A) &= \sum_{i=1}^{n} \mu_A(x_i)\, p(x_i) \\ &= \mu_A\left(x_{j1}\right) p_j\left(x_{j1}\right) + \mu_A\left(x_{j2}\right) p_j\left(x_{j2}\right) \\ &\quad + \cdots + \mu_A\left(x_{jn_j}\right) p_j\left(x_{jn_j}\right). \end{aligned} \tag{1}$$

Below we present the definition of addition of discrete fuzzy numbers suggested in [12–14, 16], where noninteractive fuzzy numbers are considered.

Definition 3 (addition of discrete fuzzy numbers [12–14, 16]). The addition of discrete fuzzy numbers $\widetilde{A}_{12} = \widetilde{A}_1 + \widetilde{A}_2$ is a discrete fuzzy number whose α-cut is given as [12–14, 16]

$$\begin{aligned} A_{12}^{\alpha} = \Big\{ x \in \left\{ \text{supp}\left(\widetilde{A}_1\right) + \text{supp}\left(\widetilde{A}_2\right) \right\} \mid \\ \min\left\{A_1^{\alpha} + A_2^{\alpha}\right\} \le x \le \max\left\{A_1^{\alpha} + A_2^{\alpha}\right\} \Big\}, \end{aligned} \tag{2}$$

where

$$\begin{aligned} &\text{supp}\left(\widetilde{A}_1\right) + \text{supp}\left(\widetilde{A}_2\right) \\ &\quad = \left\{ x_1 + x_2 \mid x_j \in \text{supp}\left(\widetilde{A}_j\right),\ j = 1, 2 \right\} \\ &\min\left\{\widetilde{A}_1^{\alpha} + \widetilde{A}_2^{\alpha}\right\} = \min\left\{ x_1 + x_2 \mid x_j \in \widetilde{A}_j^{\alpha},\ j = 1, 2 \right\} \\ &\max\left\{\widetilde{A}_1^{\alpha} + \widetilde{A}_2^{\alpha}\right\} = \max\left\{ x_1 + x_2 \mid x_j \in \widetilde{A}_j^{\alpha},\ j = 1, 2 \right\} \\ &\mu_{\widetilde{A}_1 + \widetilde{A}_2}(x) = \sup\left\{ \alpha \in [0, 1]\, x \in \widetilde{A}_1^{\alpha} + \widetilde{A}_2^{\alpha} \right\}. \end{aligned} \tag{3}$$

Definition 4 (multiplication of discrete fuzzy numbers [10, 11]). The multiplication of discrete fuzzy numbers $\widetilde{A}_{12} = \widetilde{A}_1 \cdot \widetilde{A}_2$ is a discrete fuzzy number whose α-cut is given as [10]

$$\begin{aligned} A_{12}^{\alpha} = \Big\{ x \in \left\{ \text{supp}\left(\widetilde{A}_1\right) \cdot \text{supp}\left(\widetilde{A}_2\right) \right\} \mid \\ \min\left\{A_1^{\alpha} \cdot A_2^{\alpha}\right\} \le x \le \max\left\{A_1^{\alpha} \cdot A_2^{\alpha}\right\} \Big\}, \end{aligned} \tag{4}$$

where

$$\begin{aligned} &\text{supp}\left(\widetilde{A}_1\right) \cdot \text{supp}\left(\widetilde{A}_2\right) \\ &\quad = \left\{ x_1 \cdot x_2 \mid x_j \in \text{supp}\left(\widetilde{A}_j\right),\ j = 1, 2 \right\} \\ &\min\left\{\widetilde{A}_1^{\alpha} \cdot \widetilde{A}_2^{\alpha}\right\} = \min\left\{ x_1 \cdot x_2 \mid x_j \in \widetilde{A}_j^{\alpha},\ j = 1, 2 \right\} \\ &\max\left\{\widetilde{A}_1^{\alpha} \cdot \widetilde{A}_2^{\alpha}\right\} = \max\left\{ x_1 \cdot x_2 \mid x_j \in \widetilde{A}_j^{\alpha},\ j = 1, 2 \right\} \\ &\mu_{\widetilde{A}_1 \cdot \widetilde{A}_2}(x) = \sup\left\{ \alpha \in [0, 1]\, x \in \widetilde{A}_1^{\alpha} \cdot \widetilde{A}_2^{\alpha} \right\}. \end{aligned} \tag{5}$$

Definition 5 (discrete probability distribution). The discrete probability distribution is defined as a function p where if we suppose a discrete random variable X taking K different values with probability that $X = x_i$ defined to be $P(X = x_i) = p(x_i)$, the probability $p(x_i)$ must satisfy $0 \le p(x_i) \le 1$ for each i and $\sum_{i=1}^{k} p(x_i) = 1$ [17].

Definition 6 (convolution of discrete probability distributions). Suppose X_1 and X_2 are two discrete random variables with distribution functions p_1 and p_2. The distribution function $X_1 * X_2$ is given as [17]

$$p_{12}(x) = \sum_{x = x_1 * x_2} p_1(x_1)\, p_2(x_2). \tag{6}$$

Definition 7 (a discrete Z-number [11]). A discrete Z-number is defined as an ordered pair $Z = (\widetilde{A}, \widetilde{B})$, where $\widetilde{A}$ and $\widetilde{B}$ are discrete fuzzy numbers, $\widetilde{A}$ is a fuzzy constraint on values that a random variable X may take, and $\widetilde{B}$ which has a membership function $\mu_{\widetilde{A}}$ is a fuzzy constraint on the probability measure of $\widetilde{A}$:

$$P\left(\widetilde{A}\right) \text{ is } \widetilde{B}. \tag{7}$$

The concept of a restriction has more generality than the concept of a constraint [18]. A restriction may be observed as a generalized constraint. A probability distribution is a restriction but is not a constraint [19].

Z^+-number concept is related to discrete Z-number; that is, Z^+-number is a pair of fuzzy number $\widetilde{A}$ and random number R to be defined as

$$Z^+ = \left(\widetilde{A}, R\right), \tag{8}$$

where $\widetilde{A}$ plays the same role as in discrete Z-number $Z = (\widetilde{A}, \widetilde{B})$, and R plays the role of the probability distribution P such that [10]

$$\begin{aligned} P(A) &= \sum_{i=1}^{n} \mu_A(x_i)\, p(x_i) \\ &= \mu_A\left(x_{j1}\right) p_j\left(x_{j1}\right) + \mu_A\left(x_{j2}\right) p_j\left(x_{j2}\right) \\ &\quad + \cdots + \mu_A\left(x_{jn_j}\right) p_j\left(x_{jn_j}\right). \end{aligned} \tag{9}$$

3. Computation with Discrete Z-Numbers

3.1. General Review. Zadeh has suggested a general approach for computations with Z-numbers according to Zadeh's extension principle [5]. This study is very complex in comparison with the previous one. The researchers look into using Z-numbers, but the lack of a direct and easy way to compute Z-numbers forced them to start thinking about a way to convert them into fuzzy numbers.

In [9] authors suggest an approach to convert Z-numbers into classical fuzzy numbers. They convert the second part to crisp number, but this leads to loss of original information.

The studies [4, 7, 20] are used according to what has been put forward in the study [9], but in fact this method does not give the results of high reliability. Therefore, the researchers looked for a new and simple way to calculate Z-numbers directly without conversion, based on what has been suggested in the study [5].

3.2. Addition and Multiplication of Discrete Z-Numbers. Assume $Z_1 = (\widetilde{A}_1, \widetilde{B}_1)$ and $Z_2 = (\widetilde{A}_2, \widetilde{B}_2)$ be discrete Z-numbers describing values of uncertain real valued variables X_1 and X_2. The addition and multiplication of Z-numbers $Z_{12} = Z_1 * Z_2, * \in \{+, \times\}$ are determined as follows [11]. Let $Z_1^+ = (\widetilde{A}_1, R_1)$ and $Z_2^+ = (\widetilde{A}_2, R_2)$ be given. Then

$$Z_{12}^+ = Z_1^+ * Z_2^+ = \left(\widetilde{A}_1 * \widetilde{A}_2, R_1 * R_2\right), \tag{10}$$

where R_1 and R_2 are represented by discrete probability distributions (Definition 5):

$$\begin{aligned} p_1 &= p_1(x_{11}) \setminus x_{11} + p_1(x_{12}) \setminus x_{12} + \cdots + p_1(x_{1n}) \setminus x_{1n} \\ p_2 &= p_2(x_{21}) \setminus x_{21} + p_2(x_{22}) \setminus x_{22} + \cdots + p_2(x_{2n}) \setminus x_{2n}. \end{aligned} \tag{11}$$

$\widetilde{A}_{12} = \widetilde{A}_1 * \widetilde{A}_2$ is a sum (or multiplication) of fuzzy numbers defined on the basis of Definition 3 (Definition 4) and $R_1 * R_2$ is a convolution of probability distribution defined on the basis of Definition 6.

Next, we should construct $\widetilde{B}_{12}$ by solving the following problem:

$$\mu_{\widetilde{B}_{12}}(b_{12s}) = \sup\left(\mu_{p_{12s}}(p_{12s})\right) \tag{12}$$

subject to

$$\begin{aligned} b_{12s} &= \sum_i p_{12s}(x_i)\, \mu_{\widetilde{A}_{12}}(x_i), \\ \mu_{p_{12}}(p_{12}) &= \max_{\{p_1, p_2 \mid p_{12} = p_1 \circ p_2\}} \left[\mu_{p_1}(p_1) \wedge \mu_{p_2}(p_2)\right], \\ \mu_{p_j}\left(p_j\right) &= \mu_{\widetilde{B}_j}\left(\sum_{k=1}^{n} \mu_{\widetilde{A}_j}(u_k)\, p_j(u_k)\right), \quad j = 1, 2. \end{aligned} \tag{13}$$

Thus, $Z_{12} = Z_1 * Z_2$ is obtained as $Z_{12} = (\widetilde{A}_{12}, \widetilde{B}_{12})$ [10, 11].

3.3. Ranking of Discrete Z-Numbers [11]. Ranking of discrete Z-numbers is a necessary operation in arithmetic of Z-numbers and is a challenging practical issue. Zadeh addresses the problem of ranking Z-numbers as a very important problem [5]. In contrast to real numbers, Z-numbers are ordered pairs, for ranking of which there can be no unique approach. We suggest considering comparison of Z-numbers on the basis of fuzzy optimality (FO) principle. Let Z-numbers $Z_1 = (\widetilde{A}_1, \widetilde{B}_1)$ and $Z_2 = (\widetilde{A}_2, \widetilde{B}_2)$ be given. First, it is needed to calculate the functions n_b, n_e, n_w which evaluate how much one of the Z-numbers is better, equivalent, and worse than the other one with respect to the first and the second components [11]:

$$\begin{aligned} n_b\left(Z_i, Z_j\right) &= Ps_b\left(\widetilde{\delta}_{\widetilde{A}}^{i,j}\right) + Ps_b\left(\widetilde{\delta}_{\widetilde{B}}^{i,j}\right), \\ n_e\left(Z_i, Z_j\right) &= Ps_e\left(\widetilde{\delta}_{\widetilde{A}}^{i,j}\right) + Ps_e\left(\widetilde{\delta}_{\widetilde{B}}^{i,j}\right), \\ n_w\left(Z_i, Z_j\right) &= Ps_w\left(\widetilde{\delta}_{\widetilde{A}}^{i,j}\right) + Ps_w\left(\widetilde{\delta}_{\widetilde{B}}^{i,j}\right), \end{aligned} \tag{14}$$

where $\widetilde{\delta}_{\widetilde{A}}^{i,j} = \widetilde{A}_i - \widetilde{A}_j$, $\widetilde{\delta}_{\widetilde{B}}^{i,j} = \widetilde{B}_i - \widetilde{B}_j$:

$$\begin{aligned} Ps_l\left(\widetilde{\delta}_{\widetilde{A}}^{i,j}\right) &= \frac{Poss\widetilde{\delta}_{\widetilde{A}}^{i,j} \mid n_l}{\sum_{t \in \{b,e,w\}} Poss\widetilde{\delta}_{\widetilde{A}}^{i,j} \mid n_t}, \\ Ps_l\left(\widetilde{\delta}_{\widetilde{B}}^{i,j}\right) &= \frac{Poss\widetilde{\delta}_{\widetilde{B}}^{i,j} \mid n_l}{\sum_{t \in \{b,e,w\}} Poss\widetilde{\delta}_{\widetilde{B}}^{i,j} \mid n_t}, \\ & \qquad t \in \{b, e, w\}, \end{aligned} \tag{15}$$

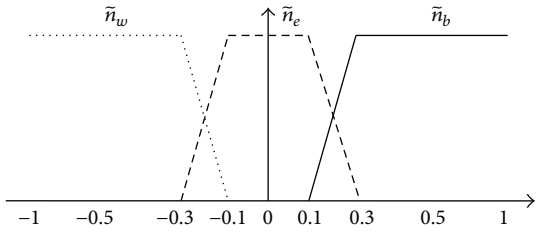


Figure 1: The membership functions of $\widetilde{n}_b$, $\widetilde{n}_e$, $\widetilde{n}_w$.

where $i, j = 1, 2$, $i \neq j$. As $\sum_{t\in\{b,e,w\}} Ps_l(\widetilde{\delta}_k^{i,j}) = 1$ always holds, one always has $n_b(Z_i, Z_j)+n_e(Z_i, Z_j)+n_w(Z_i, Z_j) = N$, where N is the number of components of a Z-number; that is, $N = 2$. The membership functions of $\widetilde{n}_b$, $\widetilde{n}_e$, $\widetilde{n}_w$ are shown in Figure 1 [11].

Next it is needed to determine the greatest k such that Z_i Pareto dominates Z_j to the degree $(1 - k)$. For this purpose, a function d is introduced:

$$d\left(Z_i, Z_j\right) = \begin{cases} 0, & \text{if } n_b\left(Z_i, Z_j\right) \leq \dfrac{2 - n_e Z_i, Z_j}{2}, \\ \left(2 \cdot n_b\left(Z_i, Z_j\right) + n_e\left(Z_i, Z_j\right) - 2\right) \times \left(n_b\left(Z_i, Z_j\right)\right)^{-1}, & \text{otherwise.} \end{cases} \tag{16}$$

Given d, the desired greatest k is found as $k = 1 - d(Z_i, Z_j)$, and then $(1 - k) = d(Z_i, Z_j)$. $d(Z_i, Z_j) = 1$ implies Pareto dominance of Z_i over Z_j, whereas $d(Z_i, Z_j) = 0$ implies no Pareto dominance of Z_i over Z_j. The degree of optimality $\mathrm{do}(Z_i)$ is determined as follows:

$$\mathrm{do}\left(Z_i\right) = 1 - d\left(Z_j, Z_i\right). \tag{17}$$

Thus, in other words, $\mathrm{do}(Z_i)$ is the degree to which one Z-number is higher than the other one. Then [11]

$$\begin{aligned} &Z_i > Z_j \quad \text{if and only if } \mathrm{do}\left(Z_i\right) > \mathrm{do}\left(Z_j\right), \\ &Z_i < Z_j \quad \text{if and only if } \mathrm{do}\left(Z_i\right) < \mathrm{do}\left(Z_j\right), \\ &\mathrm{do}\left(Z_i\right) = \mathrm{do}\left(Z_j\right), \quad \text{otherwise.} \end{aligned} \tag{18}$$

Recall that comparison of fuzzy numbers is also a matter of a degree due to related vagueness. For Z-numbers, which are more complex constructs characterized by possibilistic-probabilistic uncertainty, degree-based comparison is even more desirable.

The suggested approach may be considered as basis of a human-oriented ranking of Z-numbers. In this viewpoint, we suggest taking into account the degree of pessimism $\beta \in [0, 1]$ as a mental factor which influences a choice of a preferred Z-number. The degree of pessimism is submitted by a human observer who wishes to compare the considered Z-numbers but does not completely rely on the results obtained by the above mentioned fuzzy optimality approach. In this viewpoint, given $\mathrm{do}(Z_j) \leq \mathrm{do}(Z_i)$, we define for two Z-numbers Z_1 and Z_2 [11]

$$r\left(Z_i, Z_j\right) = \beta \mathrm{do}\left(Z_j\right) + (1 - \beta)\,\mathrm{do}\left(Z_i\right). \tag{19}$$

Then

$$\begin{aligned} &Z_i > Z_j \quad \text{if and only if } r\left(Z_i, Z_j\right) > \frac{1}{2}\left(\mathrm{do}\left(Z_i\right) + \mathrm{do}\left(Z_j\right)\right), \\ &Z_i < Z_j \quad \text{if and only if } r\left(Z_i, Z_j\right) = \frac{1}{2}\left(\mathrm{do}\left(Z_i\right) + \mathrm{do}\left(Z_j\right)\right) \end{aligned} \tag{20}$$

and $Z_i = Z_j$ otherwise [11].

The degree of pessimism β is submitted by a human being and adjusts ranking of Z-numbers to reflect human attitude to the computed do. This attitude may result from the various importance of $\widetilde{A}$ and $\widetilde{B}$ components of Z-numbers for a human being and other issues [11].

4. A Method of Decision Making under Z-Information Utility Function

Real-world decision relevant information is imprecise, uncertain, and partially reliable. Therefore, results of decision analysis based on such information are also partially reliable. This fact should prevent decision makers relying much on decision analysis results even when a very careful mathematical modeling was used.

A well-known approach to decision making under uncertainty is the use of expected utility function [21]. However, classical paradigm of expected utility function fails to express various adequate decisions due to incapability of handling imperfect decision relevant information. The extension of this paradigm to the framework of Z-information may help achieve a more realistic decision analysis technique and, at the same time, use a simple form of the utility function.

Let $\mathcal{S} = \{S_1, S_2, \ldots, S_n\}$ be a set of states of nature and let $\mathcal{X} = \{X_1, \ldots, X_l\}$, $X_k = (\widetilde{A}_k, \widetilde{B}_k)$, $k = 1, \ldots, l$, be a set of Z-valued outcomes. Denote by $\mathcal{F}_{\mathcal{S}}$ a σ-algebra of subsets of $\mathcal{S}$. Then consider $\mathcal{A} = \{f \in \mathcal{A} \mid f : \mathcal{S} \to \mathcal{X}\}$, the set of Z-valued actions, as the set of all $\mathcal{F}_{\mathcal{S}}$-measurable Z-valued functions from $\mathcal{S}$ to $\mathcal{X}$[35,32].

Linguistic information on likelihood Z_{P^l} of the states of nature is represented by Z-valued probabilities $Z_{P_i} = (\widetilde{A}_{P_i}, \widetilde{B}_{P_i})$ of the states S_i:

$$Z_{P^l} = \frac{Z_{P_1}}{S_1} + \frac{Z_{P_2}}{S_2} + \cdots + \frac{Z_{P_M}}{S_M}. \tag{21}$$

In the suggested framework, we extend a classical neo-Bayesian nomenclature as follows: elements of $\mathcal{X}$ are Z-valued outcomes; elements of $\mathcal{A}$ are Z-valued acts; elements of $\mathcal{S}$ are states of nature; elements of $\mathcal{F}_{\mathcal{S}}$ are events.

A framework of decision making with Z-information can be formalized as a 4-tuple $(\mathcal{S}, Z_{P^l}, \mathcal{X}, \mathcal{A})$. The problem of decision making with Z-valued information on the basis of EU consists in determination of an optimal act $f^* \in \mathcal{A}$: find $f^* \in \mathcal{A}$ for which $Z_{U(f^*)} \geq Z_{U(f)}$, $\forall f \in \mathcal{A}$.

Here $Z_{U(f)}$ is a Z-valued expected utility defined as

$$Z_{U(f)} = Z_{X_1} Z_{P_1} + \cdots + Z_{X_i} Z_{P_i} + \cdots + Z_{X_n} Z_{P_n}, \quad (22)$$

where multiplication and addition are defined in Section 3.2. The comparison operation $\geq$ is as defined in Section 3.3.

The suggested approach is based on direct computations with Z-numbers, without converting them to fuzzy and/or crisp numbers. This allows preserving available imprecise and partially reliable information and using it in the final comparison of alternatives.

5. Practical Applications

In this application section, we intend a problem of decision making in the field of economics. The analyzed data are obtained from Techware Incorporated in [22]. Two new software products were introduced to the market by Techware Incorporated; the company has three alternatives related to these two products: it introduces product 1 only, product 2 only, or both products. The costs for research and development for these two products are \$180,000 and \$150,000, respectively. The trend of the national economy and the consumers reaction to these products will affect the success of these products in the coming year. If the company introduces product 1, then it will have revenue of \$500,000, \$260,000, and \$120,000 for strong, fair, and weak national economy, respectively. Similarly when product 2 is introduced, there will be revenue of \$420,000, \$230,000, and 110,000 for strong, fair, and weak national economy, respectively. Finally, when introducing both products 1 and 2, the revenues will be \$820,000, \$390,000, and \$200,000 for strong, fair, and weak national economy, respectively. The experts of the company are very sure that the probabilities of strong and fair economy are about 0.30 and 0.50, respectively. The problem is to determine the best decision.

Let us proceed to formal description of the considered decision problem. The partially reliable linguistic decision relevant information in the considered problem will be described by Z-numbers. The set of alternatives is

$$\mathcal{A} = \{f_1, f_2, f_3\}, \quad (23)$$

where f_1 denotes introducing product 1, f_2 denotes introducing product 2, and f_3 denotes introducing both products (1 and 2).

The set of states of nature is

$$\mathcal{S} = \{S_1, S_2, S_3\}, \quad (24)$$

where S_1 denotes strong national economy, S_2 denotes fair national economy, and S_3 denotes weak national economy. The probabilities of states of nature are $Z_{P(S_1)} = (about\ 0.3, quite\ sure)$, $Z_{P(S_2)} = (about\ 0.5, quite\ sure)$.

The set of outcomes is

$$\begin{aligned} \mathcal{X} = \{&(low, likely), (more\ than\ low, likely), \\ &(medium, likely), \\ &(below\ compared\ to\ high, likely), \\ &(high, likely)\}. \end{aligned} \quad (25)$$

Table 1: The values of utilities for different alternatives and probabilities of states of nature.

	S_1 (about 0.3, quite sure)	S_2 (about 0.5, quite sure)	S_3 (about 0.2, quite sure)
f_1	(*High; likely*)	(*Medium; likely*)	(*Low; likely*)
f_2	(*Below compared to high; likely*)	(*Medium; likely*)	(*Low; likely*)
f_3	(*High; likely*)	(*More than low; likely*)	(*Low; likely*)

Table 2: Decision matrix with Z-number.

	S_1 Z_{41}	S_2 Z_{42}	S_3 Z_{43}
f_1	Z_{11}	Z_{12}	Z_{13}
f_2	Z_{21}	Z_{22}	Z_{23}
f_3	Z_{31}	Z_{32}	Z_{33}

The partially reliable linguistic information for the probabilities of states of nature and the utilities of each alternative taken at different states of nature is shown in Table 1.

The corresponding decision matrix with Z-number based representation is shown in Table 2.

The membership functions of the first and the second components of Z-numbers for probabilities and utilities from Table 2 are shown in Figures 2–13.

Let us proceed to solving the problem. First it is needed to determine unknown Z-number based probability $Z_{P(S_3)} = Z_{43} = (A_{43}, B_{43})$ on the basis of $Z_{P(S_1)} = Z_{41}$ and $Z_{P(S_2)} = Z_{42}$. As $Z_{P(S_3)}$ is completely determined by $Z_{P(S_1)}$ and $Z_{P(S_2)}$, its reliability B_{43} will be the same as reliabilities B_{41} and B_{42}. Therefore, to complete determination of $Z_{43} = (A_{43}, B_{43})$ it is needed to compute A_{43} on the basis of A_{41} and A_{42}. For computation of A_{43} we used the approach suggested in [19]. The determined $Z_{43} = (A_{43}, B_{43})$ is shown in Figure 13.

Based on the previous Z-number based data we compute the expected utility for each of the alternatives f_1, f_2, f_3 as follows:

$$\begin{aligned} Z_{U(f_1)} &= Z_{11} \times Z_{41} + Z_{12} \times Z_{42} + Z_{13} \times Z_{43}, \\ Z_{U(f_2)} &= Z_{21} \times Z_{41} + Z_{22} \times Z_{42} + Z_{23} \times Z_{43}, \\ Z_{U(f_3)} &= Z_{31} \times Z_{41} + Z_{32} \times Z_{42} + Z_{33} \times Z_{43} \end{aligned} \quad (26)$$

with multiplication and addition of Z-numbers described in Section 3.2.

The results of computation of expected utilities for all the alternatives are shown in Figures 14, 15, and 16.

Now determining the best alternative by comparing the computed Z-number valued utilities is needed. For comparison we will use the approach suggested in Section 3.2. In accordance with this principle, at first we obtained the degrees of optimality of the alternatives:

$$\mathrm{do}(f_1) = 1, \qquad \mathrm{do}(f_2) = 0, \qquad \mathrm{do}(f_3) = 0.92. \quad (27)$$

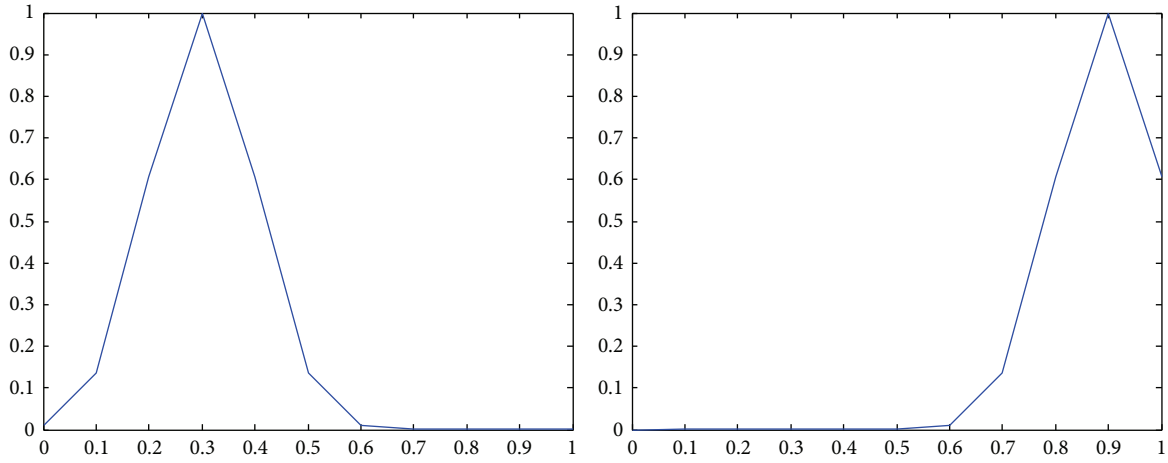


Figure 2: Representation of the first state (Z_{41}) as a Z-number.

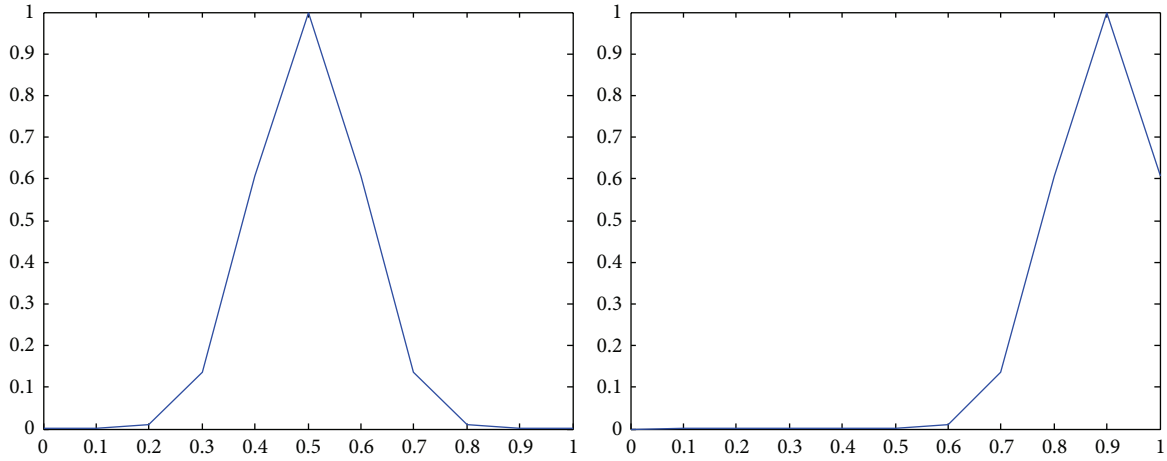


Figure 3: Representation of the second state (Z_{42}) as a Z-number.

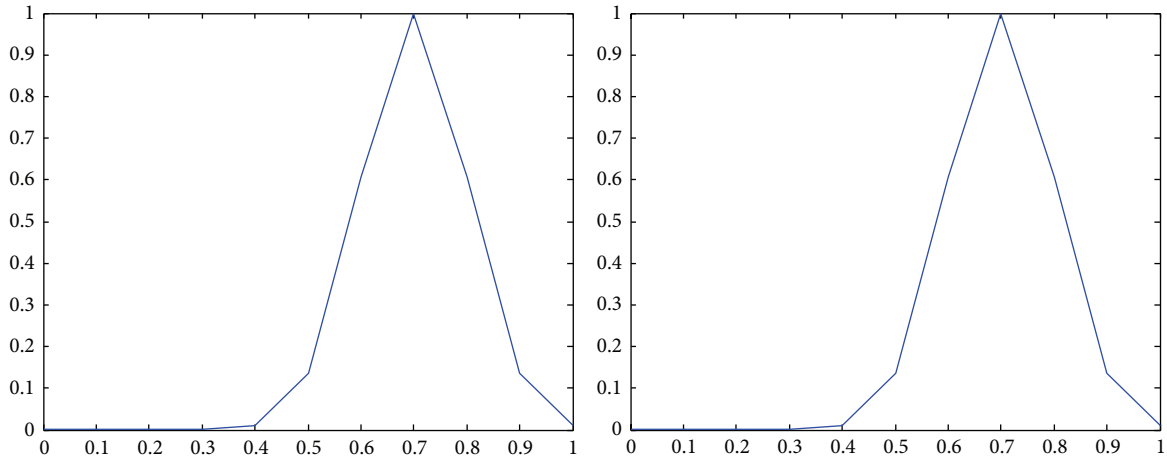


Figure 4: Representation of the first alternative in the first state (Z_{11}) as a Z-number.

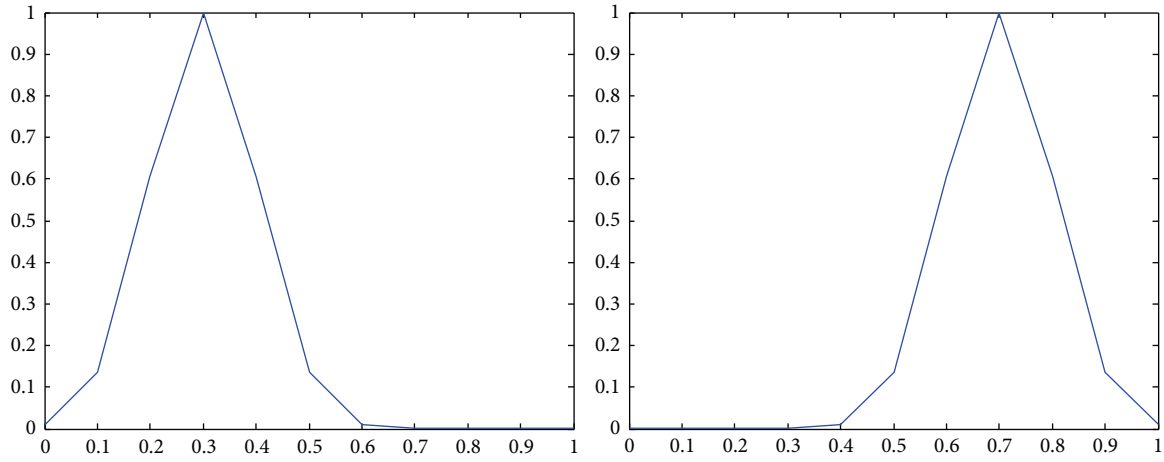


Figure 5: Representation of the first alternative in the second state (Z_{12}) as a Z-number.

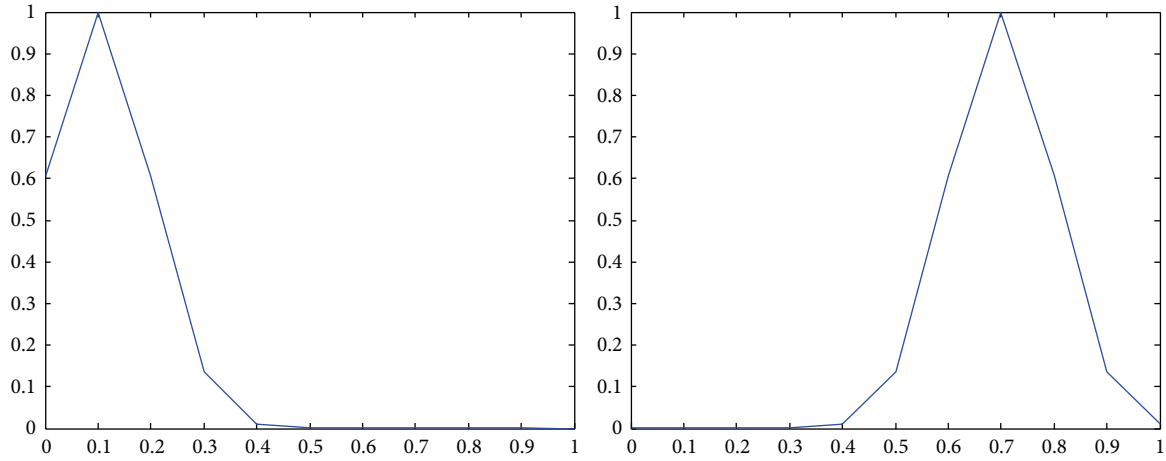


Figure 6: Representation of the first alternative in the third state (Z_{13}) as a Z-number.

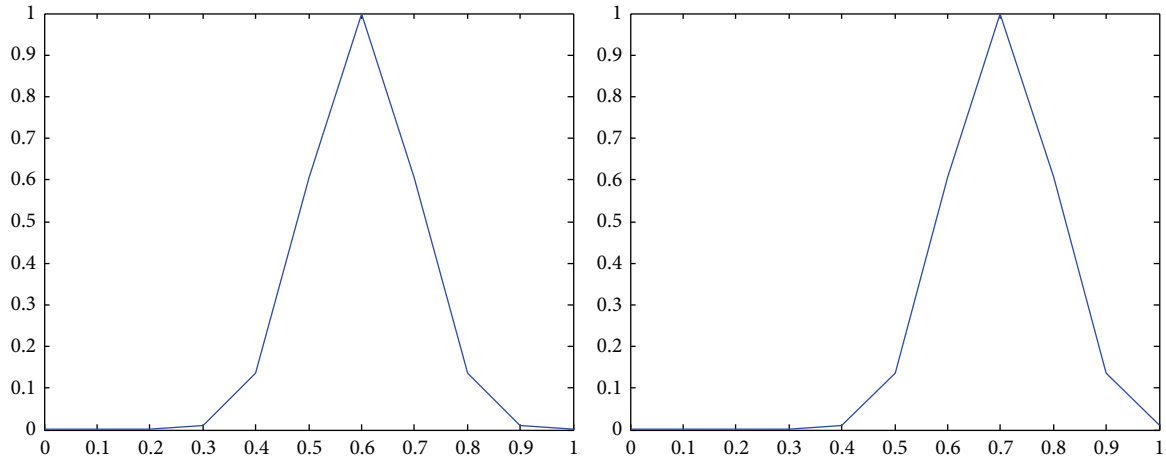


Figure 7: Representation of the second alternative in the first state (Z_{21}) as a Z-number.

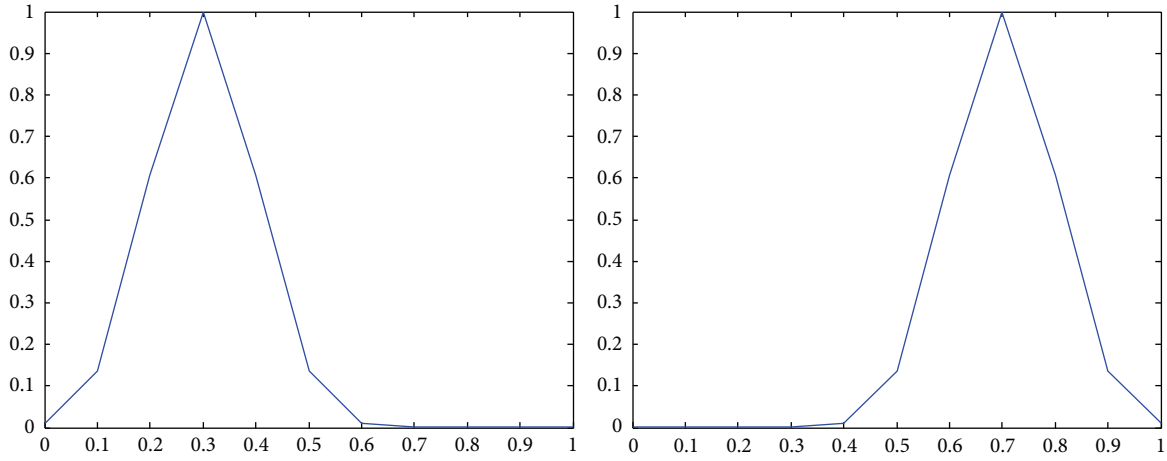


Figure 8: Representation of the second alternative in the second state (Z_{22}) as a Z-number.

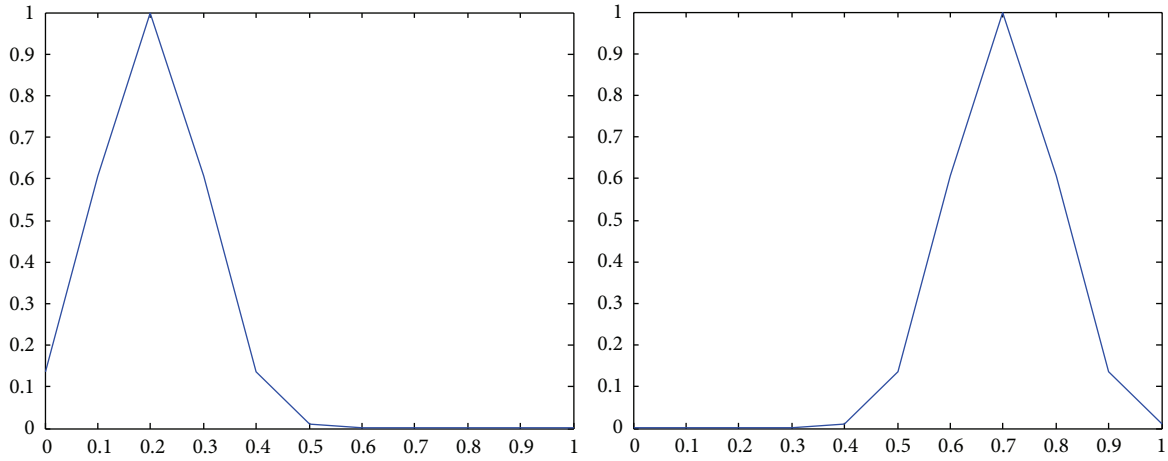


Figure 9: Representation of the second alternative in the third state (Z_{23}) as a Z-number.

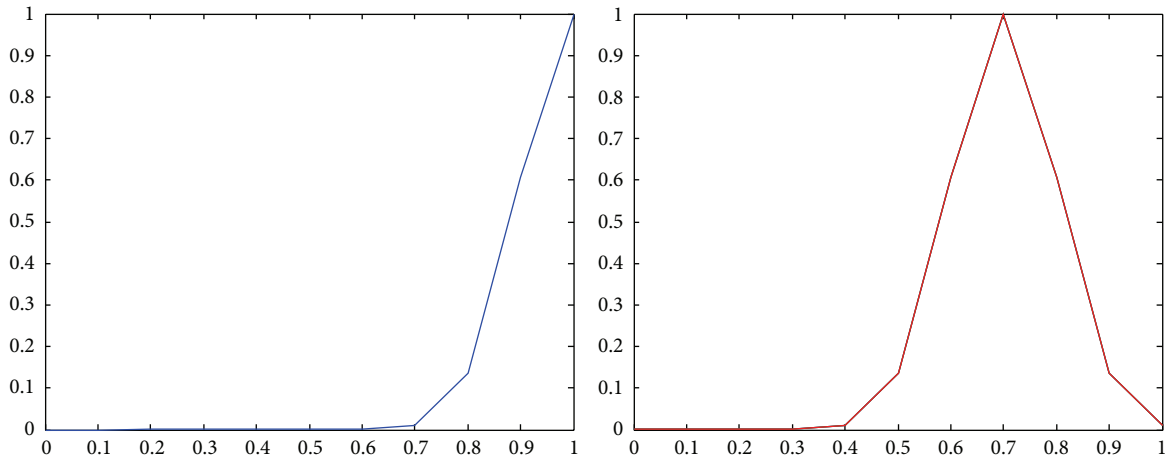


Figure 10: Representation of the third alternative in the first state (Z_{31}) as a Z-number.

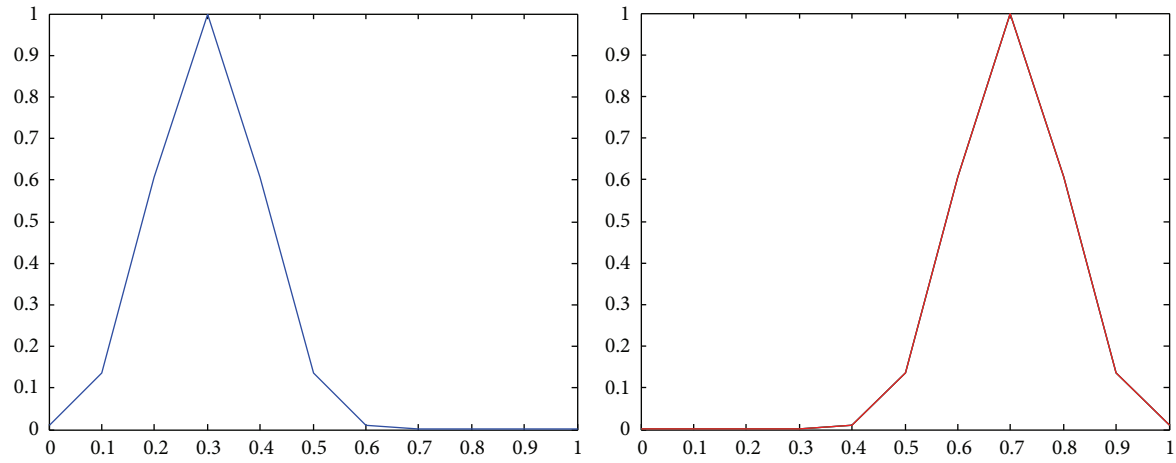


Figure 11: Representation of the third alternative in the second state (Z_{32}) as a Z-number.

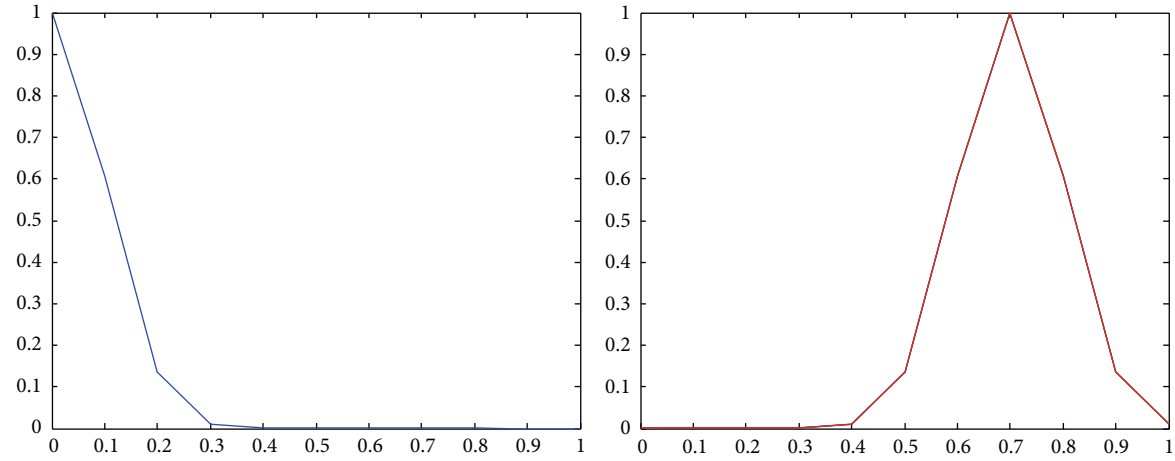


Figure 12: Representation of the third alternative in the third state (Z_{33}) as a Z-number.

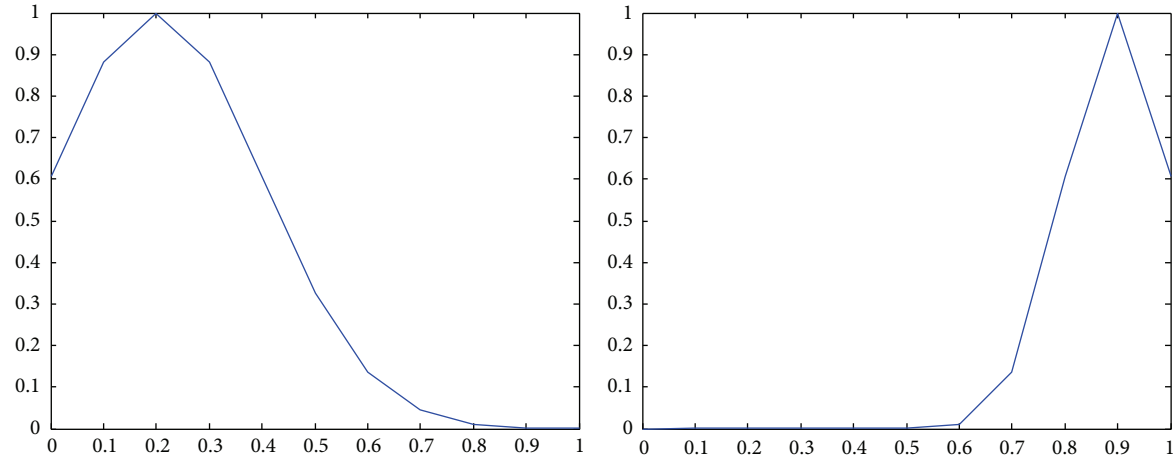


Figure 13: Representation of the first state (Z_{43}) as a Z-number.

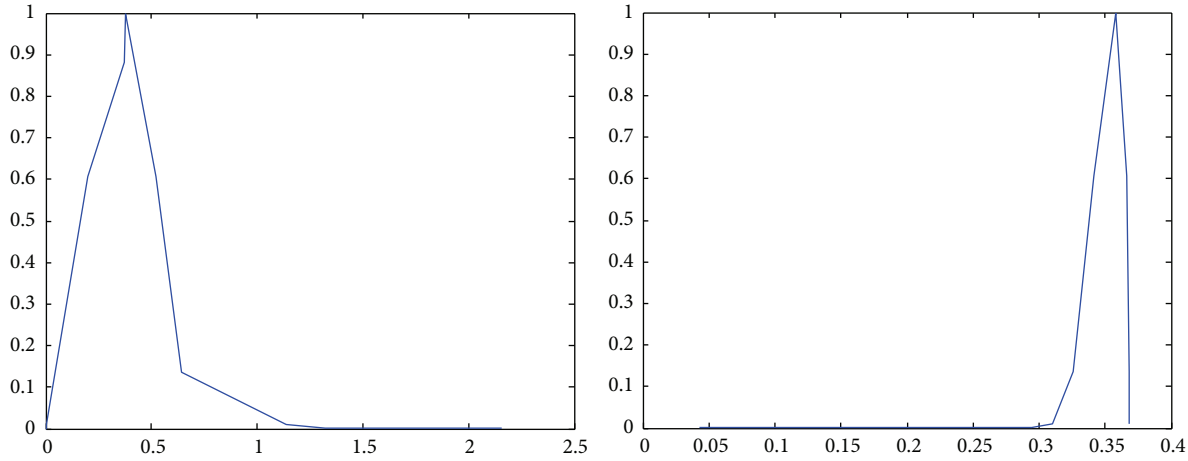


Figure 14: The expected utility results for the first alternative $Z_{U(f_1)}$.

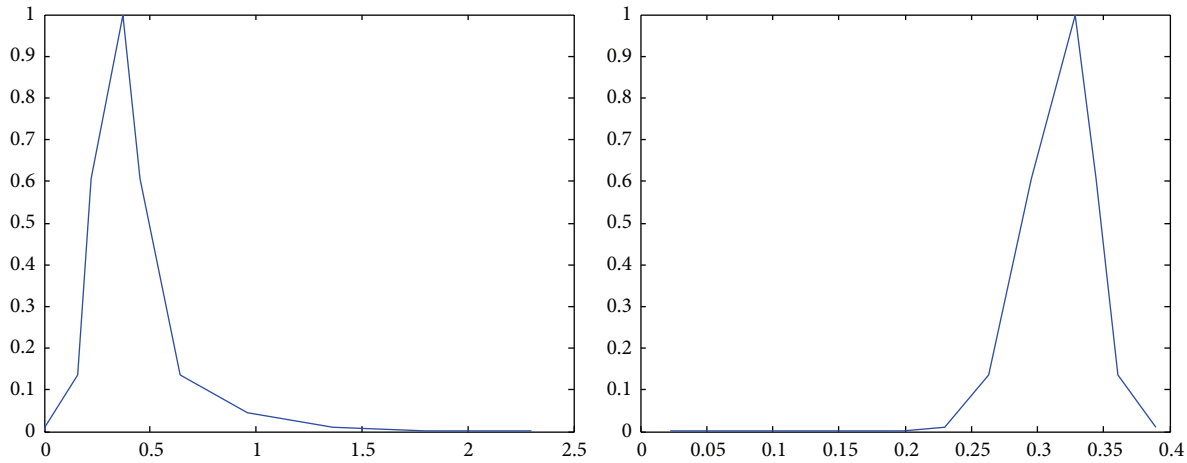


Figure 15: The expected utility results for the second alternative $Z_{U(f_2)}$.

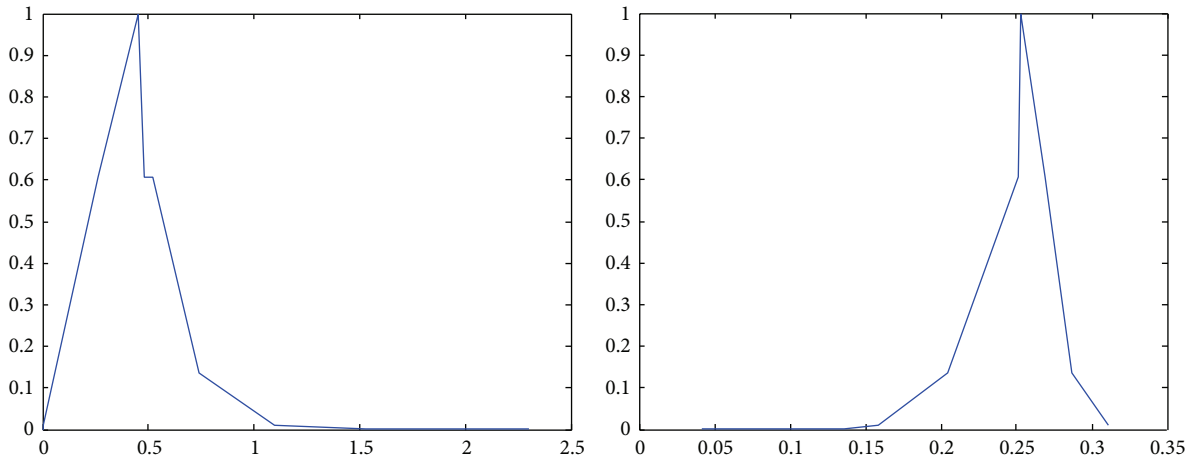


Figure 16: The expected utility results for the third alternative $Z_{U(f_3)}$.

As one can see, the second alternative is not Pareto optimal. Now it is needed to compare the first and the third alternatives. Suppose that the pessimism degree in comparison of these alternatives is $\beta = 0.3$.

Then we have

$$r\left(Z_{U(f_1)}, Z_{U(f_3)}\right) = 0.976 > \frac{1}{2}\left(\mathrm{do}\left(Z_{U(f_1)}\right) + \mathrm{do}\left(Z_{U(f_3)}\right)\right) = 0.96. \tag{28}$$

Therefore, the best action is f_1.

6. Conclusion

The concept of Z-numbers opens a door for applications in many areas, especially in decision making theory. The goal of the present study is to develop an approach for decision making under Z-information described in NL. The suggested approach utilizes the paradigm expected utility based on a direct computation of Z-numbers. The advantage of the approach is its ability to account for imprecision and partial reliability of information and relative simplicity of computations. The approach is applied to solve a benchmark decision problem in the field of economics. The obtained results show validity of the approach.

Conflict of Interests

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

References

[1] L. A. Zadeh, "Fuzzy sets," *Information and Computation*, vol. 8, no. 3, pp. 338–353, 1965.

[2] G. J. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*, Prentice Hall, Upper Saddle River, NJ, USA, 1994.

[3] S.-M. Chen, "Evaluating weapon systems using fuzzy arithmetic operations," *Fuzzy Sets and Systems*, vol. 77, no. 3, pp. 265–276, 1996.

[4] B. Kang, D. Wei, Y. Li, and Y. Deng, "Decision making using Z-numbers under uncertain environment," *Journal of Computational Information Systems*, vol. 8, no. 7, pp. 2807–2814, 2012.

[5] L. A. Zadeh, "A note on Z-numbers," *Information Sciences*, vol. 181, no. 14, pp. 2923–2932, 2011.

[6] A. V. Alizadeh, R. R. Aliev, and R. R. Aliyev, "Operational approach to Z-information-based decision making," in *Proceedings of the 10th International Conference on Application of Fuzzy Systems and Soft Computing (ICAFS '12)*, pp. 269–277, Lisbon, Portugal, August 2012.

[7] R. R. Aliev, E. K. Bodur, and D. A. T. Mraiziq, "Z-number based decision making for economic problem analysis," in *Proceedings of the 7th International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control (ICSCCW '13)*, pp. 251–257, Izmir, Turkey, September 2013.

[8] R. R. Yager, "On Z-valuations using Zadeh's Z-numbers," *International Journal of Intelligent Systems*, vol. 27, no. 3, pp. 259–278, 2012.

[9] B. Kang, D. Wei, Y. Li, and Y. Deng, "A method of converting Z-number to classical fuzzy number," *Journal of Information and Computational Science*, vol. 9, no. 3, pp. 703–709, 2012.

[10] A. V. Alizadeh, O. H. Huseynov, and R. R. Aliyev, "Numerical computations with discrete Z-numbers," in *Proceedings of the 7th International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control,(ICSCCW '13)*, pp. 71–82, Izmir, Turkey, September 2013.

[11] R. A. Aliev, A. V. Alizadeh, and O. H. Huseynov, "The arithmetic of discrete Z-numbers," *Information Sciences*, vol. 290, pp. 134–155, 2015.

[12] W. Voxman, "Canonical representations of discrete fuzzy numbers," *Fuzzy Sets and Systems*, vol. 118, no. 3, pp. 457–466, 2001.

[13] G. Wang, C. Wu, and C. Zhao, "Representation and operations of discrete fuzzy numbers," *Southeast Asian Bulletin of Mathematics*, vol. 29, no. 5, pp. 1003–1010, 2005.

[14] J. Casasnovas and J. V. Riera, "Weighted means of subjective evaluations," in *Soft Computing and Humanities in Social Sciences*, R. Seizing and V. Sanz, Eds., vol. 273 of *Studies in Fuzziness and Soft Computing*, pp. 323–345, Springer, Berlin, Germany, 2012.

[15] L. A. Zadeh, "Probability measures of fuzzy events," *Journal of Mathematical Analysis and Applications*, vol. 23, no. 2, pp. 421–427, 1968.

[16] J. Casasnovas and J. V. Riera, "On the addition of discrete fuzzy numbers," *WSEAS Transactions on Mathematics*, vol. 5, no. 5, pp. 549–554, 2006.

[17] M. Charles, J. Grinstead, and L. Snell, *Introduction to Probability*, American Mathematical Society, 1997.

[18] L. A. Zadeh, "Calculus of fuzzy restrictions," in *Fuzzy Sets and Their Applications to Cognitive and Decision Processes*, L. A. Zadeh, K. S. Fu, K. Tanaka, and M. Shimura, Eds., pp. 1–39, Academic Press, New York, NY, USA, 1975.

[19] L. A. Zadeh, "Generalized theory of uncertainty (GTU)—principal concepts and ideas," *Computational Statistics & Data Analysis*, vol. 51, no. 1, pp. 15–46, 2006.

[20] R. A. Aliev and L. M. Zeinalova, "Decision making under Z-information," in *Human-Centric Decision-Making Models for Social Sciences*, P. Guo and W. Pedrycz, Eds., vol. 502 of *Studies in Computational Intelligence*, pp. 233–252, Springer, Berlin, Germany, 2014.

[21] P. Anan, P. Pattanaik, and C. Puppe, *The Handbook of Rational and Social Choice*, Oxford University Press, 2009.

[22] W. L. Winston, S. C. Albright, and M. Broadie, *Practical Management Science*, Thomas Learning, 2nd edition, 2002.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 434263, 15 pages
http://dx.doi.org/10.1155/2015/434263

Research Article

A Method for Estimating View Transformations from Image Correspondences Based on the Harmony Search Algorithm

Erik Cuevas[1] and Margarita Díaz[2]

[1]*Departamento de Ciencias Computacionales, Universidad de Guadalajara, CUCEI , Avenida Revolución 1500, 44430 Guadalajara, JAL, Mexico*
[2]*División de Ciencia y Tecnología, Universidad de Guadalajara, CU-Norte, Carretera Federal No. 23, Km. 191, 46200 Colotlán, JAL, Mexico*

Correspondence should be addressed to Erik Cuevas; erik.cuevas@cucei.udg.mx

Received 30 September 2014; Accepted 12 December 2014

Academic Editor: Rahib H. Abiyev

In this paper, a new method for robustly estimating multiple view relations from point correspondences is presented. The approach combines the popular random sampling consensus (RANSAC) algorithm and the evolutionary method harmony search (HS). With this combination, the proposed method adopts a different sampling strategy than RANSAC to generate putative solutions. Under the new mechanism, at each iteration, new candidate solutions are built taking into account the quality of the models generated by previous candidate solutions, rather than purely random as it is the case of RANSAC. The rules for the generation of candidate solutions (samples) are motivated by the improvisation process that occurs when a musician searches for a better state of harmony. As a result, the proposed approach can substantially reduce the number of iterations still preserving the robust capabilities of RANSAC. The method is generic and its use is illustrated by the estimation of homographies, considering synthetic and real images. Additionally, in order to demonstrate the performance of the proposed approach within a real engineering application, it is employed to solve the problem of position estimation in a humanoid robot. Experimental results validate the efficiency of the proposed method in terms of accuracy, speed, and robustness.

1. Introduction

The goal of estimating geometric relations in images is to find an appropriate global transformation to overlay images of the same scene taken at different viewpoints. It can be applied in image processing when an object moves in front of a static camera and when a static scene is captured by a moving camera or multiple cameras from different viewpoints. This methodology has been widely adopted in many applications, for instance, when series of images can be stitched together to generate a panorama image [1–3]. Also, multiple image superresolution approaches can be applied in the overlapped region calculated according to the estimated geometry [4–6]. The motion of a moving object can also be estimated using its geometric relations [7] and a distributed camera network can be calibrated, where each camera's position, orientation, and focal length can be calculated based on their correspondences [8–10]. Another example is the robot position that can be controlled or estimated through the estimation of the fundamental matrix/homography [11–13].

In a modelling problem, those data that can be explained by the hypothetical model are known as *inliers* of this model. Other points, for example, those generated by matching errors, are called *outliers*. The outliers are caused by external effects not related to the investigated model. Based on different criteria, several robust techniques have been proposed to identify points as inliers or outliers, being the random sampling consensus (RANSAC) algorithm [14] the most well known [15–17].

RANSAC adopts a simple hypothesize-and-evaluation process. Under such approach, a minimal subset of elements (correspondences) is sampled randomly, and a candidate model is hypothesized using this subset. Then, the candidate model is evaluated on the entire dataset separating all

elements from the dataset into inliers and outliers, according to their degree of matching (error scale) to the candidate model. These steps are iterated until there is a high probability that an accurate model could be found during iterations. The model with the largest number of inliers is considered as the estimation result.

Although RANSAC algorithm is simple and powerful, it presents two main problems [18, 19]: the high consumption of iterations and the inflexible definition of its objective function. In the RANSAC algorithm, candidate models are generated by selecting data samples. Since such a strategy is completely random, a large number of iterations are required to explore a representative subset of noisy data and to find a reliable model that could contain the maximum number of inliers. In general terms, the number of iterations is strongly affected by the contamination level of the dataset. The other crucial issue is the objective function to evaluate the correctness of a candidate model from contaminated data. In the RANSAC methodology, the best estimation result is the model that maximizes the number of inliers. Therefore, the objective function involves the count, one by one, of the number of inliers associated with a candidate model. Such an objective function is fixed and prone to obtain suboptimal models under different circumstances [19].

Several variants have been proposed in order to enhance the performance of the RANSAC method. One example constitutes the approach MLESAC [20] which searches the best hypothesis by maximizing the likelihood via the RANSAC process by assuming that the inlier data would distribute as a Gaussian function and outliers are distributed randomly. Alternatively, instead of giving the error scale (i.e., the threshold to separate inliers from outliers) a priori, the SIMFIT method [21] proposes its prediction based on an iterative procedure. Other representative works, such as the projection-pursuit method [22] and TSSE (two-step scale estimator) [23], employ the mean shift technique to model the inlier distribution and obtain an inlier scale. Such approaches enables RANSAC to be data-driven; however, the whole process becomes quite time consuming.

Although all the proposed variants allow solving one of the two main RANSAC problems, the other challenge still remains. Such situation comes from the fact that the estimation process is approached as an optimization problem where the search strategy is a random walking algorithm while the objective function is fixed to the number of inliers associated with the candidate model. In order to overcome the typical RANSAC problems, we propose to visualize the RANSAC operation as a generic optimization procedure. Under this point of view, a new efficient search strategy can be added for reducing the number of consumed iterations. Likewise, it can be defined as a new objective function which incorporates other elements that allow an accurate evaluation of the quality of a candidate model.

Two important difficulties in selecting a search strategy for RANSAC are the high multimodality and the complex characteristics of the estimation process produced by the elevated contamination of the dataset. Under such circumstances, classical methods present a bad performance [24, 25], making way for recent new approaches that have been proposed to solve complex and ill-posed engineering problems. These methods include the application of modern optimization techniques such as evolutionary algorithms and metaheuristic techniques [26, 27] which have delivered better solutions over those obtained by classical methods.

The harmony search algorithm (HS) introduced by Geem et al. [28] is one example of these approaches. HS is an optimization algorithm based on the metaphor of the improvisation process that occurs when a musician searches for a better state of harmony. The HS produces a new candidate solution from all existing solutions. In HS, the solution vector is analogous to the harmony in music, and its generation schemes are analogous to musician's improvisations. With regard to other metaheuristics in the literature, HS imposes fewer mathematical prerequisites; therefore, it can be easily modified for solving several sorts of engineering optimization challenges [29, 30]. Numerical comparisons have established that the convergence of HS is faster than GA [29, 31, 32]. Such a fact has attracted the attention of the evolutionary computation community. It has been effectively applied to solve a wide range of practical optimization problems such as structural optimization [33], parameter estimation of the nonlinear Muskingum model [34], design optimization of water distribution networks [35], vehicle routing [36], image segmentation [37], and circle detection in images [38].

Although HS allows identifying promising regions at the solution space within a reasonable time interval, it underperforms in local searching, in particular for parameter identification applications [39–42]. In order to enhance the fine-tuning (accuracy) properties of HS, the local search parameter (BW) is dynamically adjusted to improve the balance between exploration and exploitation during the search process (see [29]). However, considering that the adjustment follows an exponential function, longer exploitation periods are allowed, affecting the exploring capacity of HS particularly when it is applied to complex objective functions. A better adjustment alternative, which employs the use of a linear model, has been recently proposed in [43]. It presents better searching capacities than the approaches based on exponential functions. For this reason, such an approach is used in our method.

In this paper, a new method is presented for the robust estimation of multiple view relations from point correspondences. The approach combines the RANSAC method with the HS. Upon such combination, the proposed method adopts a different sampling strategy in comparison to RANSAC to generate putative solutions. Under the new mechanism, new candidate solutions are built iteratively by considering the quality of models generated by previous candidate solutions, rather than relying over a pure random selection as it is the case of RANSAC. Likewise, a more accurate objective function is incorporated to accurately evaluate the quality of a candidate model. As a result, the proposed approach can substantially reduce the number of iterations still preserving the robust capabilities of RANSAC. The method is generic and its use is illustrated by the estimation of homographies, considering synthetic and real images. Additionally, in order to demonstrate the performance of the proposed approach in a real engineering application, it is

employed to solve the problem of position estimation of a humanoid robot. Experimental results validate the efficiency of the proposed method in terms of accuracy, speed, and robustness.

The paper is organized as follows. Section 2 explains the problem of image matching considering multiple views. Section 3 introduces the fundamentals of the RANSAC method. Section 4 explains the harmony search algorithm while Section 5 presents the proposed approach. Section 6 exhibits the experimental set and its performance results. Section 7 exposes a robotic application of the proposed approach. Finally, Section 8 establishes final conclusions.

2. View Relations from Point Correspondences

The problem of image matching consists in finding a geometric transformation that maps one image of a scene to another image taken from a different point of view. To determine the correspondence among points, it is necessary to find corresponding points on both images. Such point pairs can be obtained as a result of applying an automatic algorithm of detection and matching [44, 45]. The detected points are described by vectors of parameters (descriptors), and frequently these parameters do not allow discriminating one point from another with complete certainty. As a result, an erroneous matching about the correspondence of points located on different parts of different images may emerge.

In this section the geometric relations of points between two views are discussed, considering the case of homography.

Assume that there is a collection of pairs of the corresponding points that are found on two images

$$\mathbf{U} = \left\{\left(\mathbf{x}_1, \mathbf{x}_1'\right), \left(\mathbf{x}_2, \mathbf{x}_2'\right), \ldots, \left(\mathbf{x}_M, \mathbf{x}_M'\right)\right\}, \quad (1)$$

where $\mathbf{x}_i = (x_i, y_i, 1)^T$ and $\mathbf{x}_i' = (x_i', y_i', 1)^T$ are the positions of points in the first and second images, respectively.

Two perspective images can geometrically be linked through a plane Q of the scene by a homography $\mathbf{H} \in \mathfrak{R}^{3\times3}$ (see Figure 1). This projective transformation $\mathbf{H}$ relates corresponding points of the plane projected into two images by $\mathbf{x}_i' = \mathbf{H}\mathbf{x}_i$ or $\mathbf{x}_i = \mathbf{H}^{-1}\mathbf{x}_i'$. The homography across two views can be computed by solving a linear system from a set of four point matches [46]. The quality of the estimated homography $\mathbf{H}$ is evaluated by considering the distance between the position of the point calculated with the help of the matrix $\mathbf{H}$ and the actually observed position. Therefore, the mismatch error EH_i^2 produced by the i-correspondence $(\mathbf{x}_i, \mathbf{x}_i')$ is defined as the sum of squared distances from the points to their estimated positions:

$$EH_i^2 = \left[d\left(\mathbf{x}_i', \mathbf{H}\mathbf{x}_i\right)\right]^2 + \left[d\left(\mathbf{x}_i, \mathbf{H}^{-1}\mathbf{x}_i'\right)\right]^2, \quad (2)$$

where $\eta = d(\mathbf{x}_i, \mathbf{H}^{-1}\mathbf{x}_i')$ and $\eta' = d(\mathbf{x}_i', \mathbf{H}\mathbf{x}_i)$ correspond to the errors produced in the first and second images, respectively.

Figure 2 shows the error evaluation process of $\mathbf{H}$ for a particular example which involves five correspondences $\mathbf{U} = \{(\mathbf{x}_1, \mathbf{x}_1'), \ldots, (\mathbf{x}_5, \mathbf{x}_5')\}$ distributed in both views. In the example, the correspondence $(\mathbf{x}_3, \mathbf{x}_3')$ presents a considerable error evaluated as the distances (η, η') between the points $(\mathbf{x}_3, \mathbf{x}_3')$ and their positions calculated with the use of the matrix $\mathbf{H}$ $(\mathbf{H}\mathbf{x}_3, \mathbf{H}^{-1}\mathbf{x}_3')$.

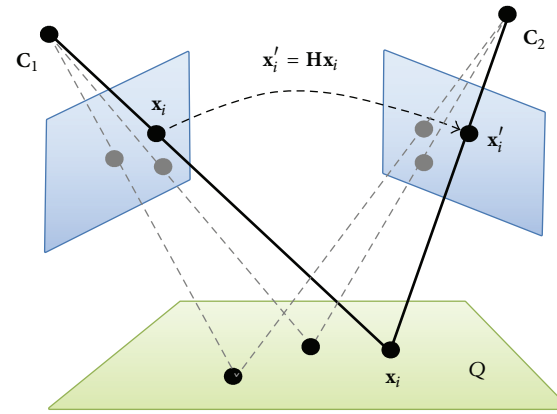


Figure 1: Homography from a plane between two views.

3. Random Sampling Consensus (RANSAC) Algorithm

The goal of RANSAC is to estimate the geometric transformation (the homography $\mathbf{H}$) from image correspondences over two views. Potentially there are a significant number of mismatches amongst the correspondences. Correct matches will obey the homography transformation. Therefore, the aim is to obtain a set of *inliers* consistent with the homography transformation by using a robust technique. In this case *outliers* are points inconsistent with the homography transformation. In order to solve such a problem, the RANSAC algorithm has proven to be the most successful [15–17].

RANSAC solves the problem of model parameters estimation by finding the best hypothesis h^B among the set of all possible hypotheses H generated by the source data. Such source data are typically contaminated by noise. In order to build the hypothesis h_i about the unknown parameters, a sample $\mathbf{S}_i$ of the minimum size (s) required for model estimation is obtained (e.g., a sample of only two points is sufficient to calculate a straight line, $s = 2$, and of four to obtain a homography, $s = 4$). Under this consideration, the probability of finding an outlier is reduced. Considering that the number of elements contained in a sample is small, the amount of possible samples that can be generated from the complete source data $\mathbf{U}$ is enormous. Under such circumstances, the exhausting testing of all samples for a reasonable time is impossible. RANSAC faces such problem because it only considers G samples which are randomly selected and evaluated. Algorithms of the RANSAC family consist of G iterations of the following cycle.

(1) Construct a sample $\mathbf{S}_i \subset \mathbf{U}$ consisting of s different elements.

(2) Build the hypothesis h_i based on the sample $\mathbf{S}_i$.

(3) Evaluate the degree of agreement A_i of the hypothesis h_i with the set of all source data $\mathbf{U}$.

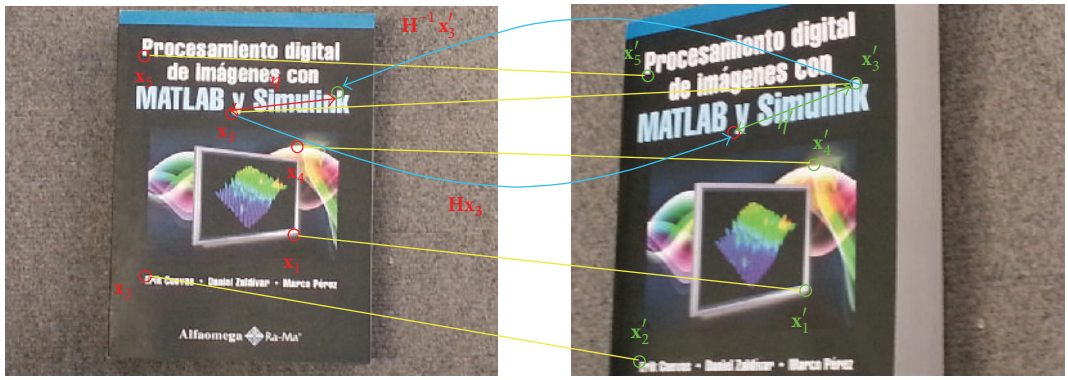


FIGURE 2: Example of evaluation process for a particular homography **H**.

After the construction and evaluation of all G hypotheses, the hypothesis h^B with the best degree of agreement is chosen among them. It is considered as a robust estimate of the model parameters. Such operation can be described as follows:

$$h^B = \arg\max_{i=1,\ldots,G} A_i(\mathbf{U}, h_i). \tag{3}$$

The maximization of the degree of agreement (number of inliers) is equivalent to the minimization of the penalty function whose value depends on the number of outliers. Therefore, the degree of agreement $A_i(\mathbf{U}, h_i)$ is computed as follows:

$$A_i(\mathbf{U}, h_i) = \sum_{j=1}^{M} \theta\left(e_j^2(h_i)\right), \quad j = 1, \ldots, M,$$
$$\theta\left(e_j^2(h_i)\right) = \begin{cases} 0 & e_j^2(h_i) > \mathrm{Th} \\ 1 & e_j^2(h_i) \leq \mathrm{Th}, \end{cases} \tag{4}$$

where Th is a permissible error, M is the number of elements contained in the source data $\mathbf{U}$, and $e_j^2(h_i)$ is the quadratic error produced by the jth data considering the hypothesis h_i. In the context of this paper, $e_j^2(h_i)$ corresponds to EH_i^2 which represents the error produced by the ith correspondence.

The hypothesis with a minimum penalty (i.e., with the maximum degree of agreement) is chosen as the best matching criterion. In the original scheme of RANSAC, the quality of a hypothesis is defined as the number of inliers. For a given value of the permissible error Th, the point j that produces the error $e_j^2(h_i)$ is regarded to be an inlier of h_i if its value does not exceed the threshold Th; otherwise the point is regarded as an outlier.

In the RANSAC algorithm, the optimal hypothesis h^B is found and the penalty is minimized by using a search strategy of random walking; therefore many attempts are necessary to investigate in sufficient detail the space of possible samples and to find the sample for which the hypothesis has the greatest degree of agreement on the source data. The number of iterations and thus the time spent for the search can be reduced by choosing points according to some directed rules, rather than randomly. Optimization algorithms can be considered as a robust scheme in contrast to the random search [47]. In an optimization algorithm, new candidate solutions are generated in accordance to the information obtained from past candidate solutions.

In this paper, we propose a different approach based on the HS as optimization algorithm. The goal is to demonstrate that the new method, by combining the idea of testing minimum-sized samples with the directed search inspired by the improvisation process that occurs when a musician searches for a better state of harmony, allows performing an efficient search among the correspondences to generate models of higher quality. It is also shown that the number of inliers found by the new method with the use of a fixed number of samples is significantly greater than the number of inliers determined by the family of algorithms based on RANSAC.

4. Harmony Search Algorithm

In the basic HS, each solution is called a "harmony" and is represented by an n-dimension real vector. An initial population of harmony vectors are randomly generated and stored within a harmony memory (HM). A new candidate harmony is thus generated from the elements in the HM by using a memory consideration operation either by a random reinitialization or a pitch adjustment operation. Finally, the HM is updated by comparing the new candidate harmony and the worst harmony vector in the HM. The worst harmony vector is replaced by the new candidate vector in case it is better than the worst harmony vector in the HM. The above process is repeated until a certain termination criterion is met. The basic HS algorithm consists of three basic phases: HM initialization, improvisation of new harmony vectors, and updating of the HM. The following discussion addresses details about each stage.

4.1. Initializing the Problem and Algorithm Parameters. In general, the global optimization problem can be summarized as follows: $\min f(\mathbf{p}) : p(j) \in [l(j), u(j)]$, $j = 1, 2, \ldots, n$, where $f(\mathbf{p})$ is the objective function, $\mathbf{p} = (p(1), p(2), \ldots, p(n))$ is the set of design variables, n is the number of design variables, and $l(j)$ and $u(j)$ are the lower and upper bounds for the

design variable $p(j)$, respectively. The parameters for HS are the harmony memory size, that is, the number of solution vectors lying on the harmony memory (HM), the harmony-memory consideration rate (HMCR), the pitch adjusting rate (PAR), the distance bandwidth (BW), and the number of improvisations (NI) which represents the total number of iterations. It is obvious that an adequate selection for HS parameters would enhance the algorithm's ability to search for the global optimum under a high convergence rate.

4.2. Harmony Memory Initialization. In this stage, initial vector components at HM, that is, HMS vectors, are configured. Let $\mathbf{p}_i = \{p_i(1), p_i(2), \ldots, p_i(n)\}$ represent the ith randomly generated harmony vector: $p_i(j) = l(j) + (u(j) - l(j)) \cdot \text{rand}(0,1)$ for $j = 1, 2, \ldots, n$ and $i = 1, 2, \ldots, \text{HMS}$, where rand(0,1) is a uniform random number between 0 and 1. Then, the HM matrix is filled with the HMS harmony vectors as follows:

$$\text{HM} = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \vdots \\ \mathbf{p}_{\text{HMS}} \end{bmatrix}. \tag{5}$$

4.3. Improvisation of New Harmony Vectors. In this phase, a new harmony vector $\mathbf{p}_{\text{new}}$ is built by applying the following three operators: memory consideration, random reinitialization, and pitch adjustment. Generating a new harmony is known as "improvisation." In the memory consideration step, the value of the first decision variable $p_{\text{new}}(1)$ for the new vector is chosen randomly from any of the values already existing in the current HM, that is, from the set $\{p_1(1), p_2(1), \ldots, p_{\text{HMS}}(1)\}$. For this operation, a uniform random number r_1 is generated within the range $[0, 1]$. If r_1 is less than HMCR, the decision variable $p_{\text{new}}(1)$ is generated through memory considerations; otherwise, $p_{\text{new}}(1)$ is obtained from a random reinitialization between the search bounds $[l(1), u(1)]$. Values of the other decision variables $p_{\text{new}}(2), p_{\text{new}}(3), \ldots, p_{\text{new}}(n)$ are also chosen accordingly. Therefore, both operations, memory consideration and random reinitialization, can be modelled as follows:

$$p_{\text{new}}(j) = \begin{cases} p_i(j) \in \{p_1(j), p_2(j), \ldots, p_{\text{HMS}}(j)\} \\ \quad \text{with probability HMCR} \\ l(j) + (u(j) - l(j)) \cdot \text{rand}(0,1) \\ \quad \text{with probability } 1 - \text{HMCR}. \end{cases} \tag{6}$$

Every component obtained by memory consideration is further examined to determine whether it should be pitch-adjusted. For this operation, the pitch adjusting rate (PAR) is defined as to assign the frequency of the adjustment and the bandwidth factor (BW) to control the local search around the selected elements of the HM. Hence, the pitch adjusting decision is calculated as follows:

$$p_{\text{new}}(j) = \begin{cases} p_{\text{new}}(j) = p_{\text{new}}(j) \pm \text{rand}(0,1) \cdot \text{BW} \\ \quad \text{with probability PAR} \\ p_{\text{new}}(j) \quad \text{with probability } (1 - \text{PAR}). \end{cases} \tag{7}$$

Pitch adjusting is responsible for generating new potential harmonies by slightly modifying original variable positions. Such operation can be considered similar to the mutation process in evolutionary algorithms. Therefore, the decision variable is either perturbed by a random number between $-\text{BW}$ and BW or left unaltered. In order to protect the pitch adjusting operation, it is important to assure that points lying outside the feasible range $[l, u]$ must be reassigned, that is, truncated to the maximum or minimum value of the interval.

4.4. Updating the Harmony Memory. After a new harmony vector $\mathbf{p}_{\text{new}}$ is generated, the harmony memory is updated by the survival of the fit competition between $\mathbf{p}_{\text{new}}$ and the worst harmony vector $\mathbf{p}_w$, according to its fitness value, in the HM. Therefore $\mathbf{p}_{\text{new}}$ will replace $\mathbf{p}_w$ and become a new member of the HM in case the fitness value of $\mathbf{p}_{\text{new}}$ is better than the fitness value of $\mathbf{p}_w$.

4.5. Computational Procedure. The computational procedure of the basic HS can be summarized as shown in Procedure 1 [18].

4.6. Dynamical Linear Adjustment of BW. Every metaheuristic algorithm needs to address the issue of exploration-exploitation of the search space. Exploration is the process of visiting entirely new points of a search space whilst exploitation is the process of refining those points within the neighborhood of previously visited locations in order to improve their solution quality.

In HS, the BW parameter controls the local search around HM elements. A large BW value eases the algorithm's searching at a larger scope, while a small BW value is appropriate for fine-tuning of best solution vectors.

In the standard HS, the BW value is considered as a constant number. However, in this work, the BW value is dynamically adjusted as to favor exploration at early stages while exploitation is reinforced during final stages of the searching process. The adjustment uses a linear model defined as follows:

$$\text{BW}(k) = \begin{cases} \text{BW}_{\max} - \left(\dfrac{\text{BW}_{\max} - \text{BW}_{\min}}{2 \cdot \text{NI}}\right) \cdot 3k & \text{if } k < \left(\dfrac{2}{3}\right)\text{NI} \\ \text{BW}_{\min} & \text{if } k \geq \left(\dfrac{2}{3}\right)\text{NI}, \end{cases} \tag{8}$$

where k is the iteration index, while $\text{BW}_{\max}$ and $\text{BW}_{\min}$ are the maximum and minimum BW values, respectively. In contrast to exponential adjustment [26], linear models, as the one used in this paper, allow a better balance between exploration and exploitation (fine-tuning) of the search process [40].

Since all candidate solutions are generated by using the HS operators, there is a low probability to be trapped into local minima [48]. HS can effectively handle challenging multimodal optimization problems [49, 50]. Such fact contrasts to well-known genetic algorithms (GA) [51] and particle swarm optimization (PSO) [52] which usually tends

Step 1. Set the parameters HMS, HMCR, PAR, BW and NI.
Step 2. Initialize the HM and calculate the objective function value of each harmony vector.
Step 3. Improvise a new harmony $\mathbf{p}_{\text{new}}$ as follows:
for ($j = 1$ to n) do
if ($r_1 <$ HMCR) then
Select randomly a number a where $a \in (1, 2, \ldots, \text{HMS})$
$p_{\text{new}}(j) = p_a(j)$
if ($r_2 <$ PAR) then
$p_{\text{new}}(j) = p_{\text{new}}(j) \pm r_3 \cdot \text{BW}$ where $r_1, r_2, r_3 \in \text{rand}(0, 1)$
end if
if $p_{\text{new}}(j) < l(j)$
$p_{\text{new}}(j) = l(j)$
end if
if $p_{\text{new}}(j) > u(j)$
$p_{\text{new}}(j) = u(j)$
end if
else
$p_{\text{new}}(j) = l(j) + r \cdot (u(j) - l(j))$, where $r \in \text{rand}(0, 1)$
end if
end for
Step 4. Update the HM as $\mathbf{p}_w = \mathbf{p}_{\text{new}}$ if $f(\mathbf{p}_{\text{new}}) > f(\mathbf{p}_w)$
Step 5. If NI is completed, the best harmony vector $\mathbf{p}_b$ according to its fitness value in the HM is returned; otherwise go back to *Step 3*.

Procedure 1

to conduct the whole population towards the best candidate solution [53] producing premature convergence.

5. Method for Geometric Estimation Using HS

The estimation of model parameters in algorithms of the RANSAC family is implied to find an optimal sample of length s from a set consisting of M elements. In the standard scheme, RANSAC uses a random walking algorithm as a search strategy. The idea of the proposed method considers the use of HS to generate samples based on information about their quality, rather than randomness. The quality of a sample, that is, the fitness of a harmony $f(\mathbf{p}_i)$, is defined as the matching degree of the hypothesis h_i that is constructed based on the correspondence numbers coded within $\mathbf{p}_i$.

Considering that the problem consists in estimating the parameters of $\mathbf{H}$ through a set $\mathbf{U} = \{(\mathbf{x}_1, \mathbf{x}'_1), (\mathbf{x}_2, \mathbf{x}'_2), \ldots, (\mathbf{x}_M, \mathbf{x}'_M)\}$ of M different correspondences, the proposed approach can be described as shown in Algorithm 1.

The proposed approach combines the RANSAC method with the HS adopting a different sampling strategy in comparison to RANSAC to generate putative solutions. Under the new mechanism, at each iteration, new candidate solutions are built taking into account the quality of the models that have been generated by previous candidate solutions, rather than purely random as it is the case in RANSAC.

Since the approach visualizes the RANSAC operation as a generic optimization procedure, different objective functions can be incorporated to accurately evaluate the quality of a candidate model. Although several objective functions can be tested, this work employs the expression in Equation (A).

In contrast to the traditional RANSAC algorithm, the objective function considers two different aims: the number of inliers and the approximation error. The idea is to find the candidate homography that maximizes the number of inliers and simultaneously minimizes the approximation error. Under such circumstances, the obtained estimation represents the solution that presents the best trade-off between both objectives. As a result, the proposed approach can substantially reduce the number of iterations, still preserving the robust capabilities of RANSAC method.

6. Experimental Results

In this section, a comprehensive set of experiments have been conducted to test the performance of the proposed approach. The results are divided into two different categories: (1) effect of the main HS parameters in the estimation results and (2) comparison results over synthetic and real homographies.

In the experiments, three performance indexes are considered: the number of inliers (NofI), the error (E_s, E_r), and the number of function evaluations (NFE). The first two indexes assess the accuracy of the solution whereas the last one measures the computational cost.

The number of inliers (NofI) expresses the amount of elements contained in the set $\mathbf{I}$ of detected inliers. The error (E_s, E_r) provides a quality measure of the estimated relation. In case of synthetic data, the error is calculated as

$$E_s = \left(\sum_{ij} \frac{d^2(\mathbf{x}_i^j, \hat{\mathbf{x}}_i^j)}{\text{NofI}} \right)^{1/2}, \quad i \in \mathbf{I}, \; j \in \{1, 2\}, \tag{9}$$

(1) Configuration

(a) Set the parameters HMS, HMCR, PAR, BW_{min}, BW_{max} and NI.

(2) Initial population.

(a) Build the harmony memory (HM) $HM = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_{HMS}\}$ where each individual $\mathbf{p}_i$ consists 4 random non-repeating indices from 1 to M.

(b) Compute homography $\mathbf{H}_i$ (hypothesis h_i) by using the indices from $\mathbf{p}_i$.

(c) Calculate the fitness value $f(\mathbf{p}_i)$ as the matching quality of the constructed homography $\mathbf{H}_i$ considering the whole available data $\mathbf{U}$. Such fitness value is calculated by using a new objective function defined as:

$$F\left(\mathbf{a}_i(k)\right) = \sum_{j=1}^{M} \theta\left(e_j^2(h_i)\right) - \lambda \cdot e_j^2(h_i) \qquad \theta\left(e_j^2(h_i)\right) = \begin{cases} 0 & e_j^2(h_i) > \mathrm{Th} \\ 1 & e_j^2(h_i) \leq \mathrm{Th} \end{cases}, \tag{A}$$

where $e_j^2(h_i)$ represents the quadratic error produced by the jth correspondence considering the hypothesis h_i whereas λ is the penalty associated with the mismatch magnitude. Such error corresponds to the mismatch EH_j^2 generated by the evaluation of $\mathbf{H}_i$.

(3) Iterations $k = 1, \ldots, \mathrm{NI}$.

(a) Generate a new harmony $\mathbf{p}_{new}$ (candidate solution) as follows:

$$\mathrm{BW}(k) = \begin{cases} \mathrm{BW}_{max} - \left(\dfrac{\mathrm{BW}_{max} - \mathrm{BW}_{min}}{2 \cdot \mathrm{NI}}\right) \cdot 3k & \text{if } k < \left(\dfrac{2}{3}\right)\mathrm{NI} \\ \mathrm{BW}_{min} & \text{if } k \geq \left(\dfrac{2}{3}\right)\mathrm{NI} \end{cases}$$

for ($j = 1$ to n) do
  if ($r_1 <$ HMCR) then
    Select randomly a number a where $a \in (1, 2, \ldots, \mathrm{HMS})$
    $p_{new}(j) = p_a(j)$
    if ($r_2 <$ PAR) then
      $p_{new}(j) = p_{new}(j) \pm r_3 \cdot \mathrm{BW}(k)$ where $r_1, r_2, r_3 \in \mathrm{rand}(0, 1)$
    end if
    if $p_{new}(j) < l(j)$
      $p_{new}(j) = l(j)$
    end if
    if $p_{new}(j) > u(j)$
      $p_{new}(j) = u(j)$
    end if
  else
    $p_{new}(j) = l(j) + r \cdot (u(j) - l(j))$, where $r \in \mathrm{rand}(0, 1)$
  end if
end for

(b) Compute homography $\mathbf{H}_i$ by using the indices from $\mathbf{p}_{new}$.

(d) Calculate the fitness value $f(\mathbf{p}_{new})$ as the matching quality of the constructed homography $\mathbf{H}_{new}$ considering the whole available data $\mathbf{U}$. Such fitness value is calculated by using the objective function described in (A).

(e) Update the HM as $\mathbf{p}_w = \mathbf{p}_{new}$ if $f(\mathbf{p}_{new}) > f(\mathbf{p}_w)$

(4) Estimation result

(a) The best estimation $\mathbf{H}^{\mathbf{B}}$ consists of the parameters computed by using the indices from the best element $\mathbf{p}^{\mathbf{B}}$ of HMin terms of its affinity, so that $\mathbf{p}^B = \mathrm{argmax}_{i=1,\ldots,\mathrm{HMS}} f(\mathbf{p}_i)$.

Algorithm 1

where $\mathbf{x}_i^j$ is the inlier point calculated by the estimated relation in the j-view, $\hat{\mathbf{x}}_i^j$ is the inlier ground true point, and $d(\cdot)$ is the Euclidian distance between the points. Therefore, E_s evaluates the fit of the estimated relation, computed from the noisy data, against the known ground truth points.

In the case of real data the error is assessed from the standard deviation of the inliers. Thus, E_r is computed as follows:

$$E_r = \left(\sum_i \frac{e_i^2}{\mathrm{NofI}}\right)^{1/2}, \quad i \in \mathbf{I}, \tag{10}$$

Table 1: Effect of the HS parameters in the estimation process.

	(NofI, E_s)			
	HMCR = 0.5	HMCR = 0.6	HMCR = 0.7	HMCR = 0.8
PAR = 0.1	(21, 4.2147)	(26, 3.8124)	(29, 3.4721)	(27, 4.0112)
PAR = 0.2	(22, 4.1457)	(35, 2.0974)	(36, 2.1474)	(31, 3.3784)
PAR = 0.3	(21, 4.5714)	(38, 1.1124)	**(40, 0.8514)**	(35, 2.0053)
PAR = 0.3	(23, 4.0781)	(34, 2.0078)	(37, 2.0012)	(31, 3.4079)

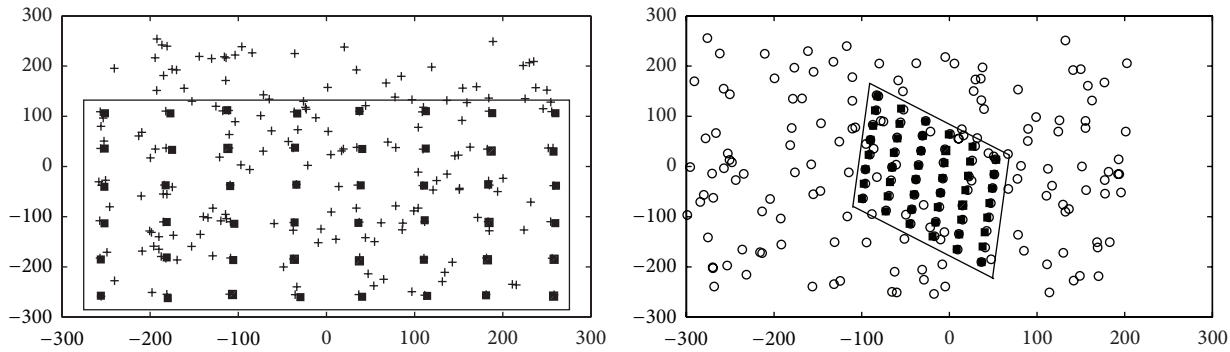


Figure 3: A test example where the HS-RANSAC has been applied to estimate a random transformation **H** considering only the 75% of additional outliers. (a) The first view and (b) the second view, with black squares representing the detected inliers.

where e_i^2 is the quadratic error produced by the ith inlier. In the context of this paper, e_i^2 corresponds to EH_i^2 which represents the error produced by the ith inlier.

The number of function evaluations (NFE) specifies the total number of transformations that have been evaluated by the algorithm until the best estimation has been reached.

6.1. Effect of the HS Parameters. Several parameters define the performance of HS. However, from all of them, the harmony-memory consideration rate (HMCR) and pitch adjusting rate (PAR) are the most important [55]. To study the impact of these parameters, over the performance of HS in the estimation procedure, different values have been tested on the computation of a synthetic homography. Such a homography was generated, in its first view, by using a rectangular pattern of 8×6 elements within a 2-dimensional space of $[-300, 300]$. Then, such points were transformed by a random homography **H** and contaminated by normally distributed noise for constructing their correspondences in the second view. A set of outliers was added by selecting randomly data points within the space limits. In the test, the fraction of outliers is of 75%. In order to illustrate the experimental setup, Figures 3(a) and 3(b) exhibit the first and second views, respectively. Considering the correspondence points, the HS-RANSAC algorithm generates the estimation of **H**. In Figure 3(a), the black squares indicate the position in the first view of a point from the second view as a result of the **H** transformation. Likewise, the black squares in Figure 3(b) exhibit the position in the second view of a point from the first view as a result of the **H** transformation.

Table 2: HS-RANSAC estimator parameters.

HMS	HMCR	PAR	BW_{max}	BW_{min}	NI	λ	Th
50	0.7	0.3	10	1	950	0.001	5

In the experiment, the maximum number of iterations is set to 950. HMS, BW_{max}, BW_{min}, λ, and Th are fixed to 50, 10, 1, 0.001, and 5, respectively. The results report the number of inliers (NofI) and the produced estimation error (E_s) of HS-RANSAC, averaged over 30 runs, for the different values of HMCR and PAR. In the experiment, the parameter values are modified considering specific interval. HMCR varies from 0.5 to 0.8 whereas PAR changes from 0.1 to 0.4. The results, shown in Table 1, suggest that a proper combination of different parameter values can improve the performance of HS-RANSAC and the quality of the estimations. The best parameter configuration in the experiment is highlighted in Table 1.

After considering the analysis of Table 1, the parameter values for the proposed estimator are defined in Table 2. Once defined, such values have been kept in all experiments reported in this paper.

6.2. Comparison Results over Synthetic and Real Homographies. We have applied the proposed method to estimate homographies on real and synthetic data in order to compare its performance against other estimation algorithms such as the standard RANSAC [14], the MLESAC [20], the SIMFIT method [21], the projection-pursuit algorithm [22], the TSSE [23], and the PSO algorithm (PSO-RANSAC) [54]. The first

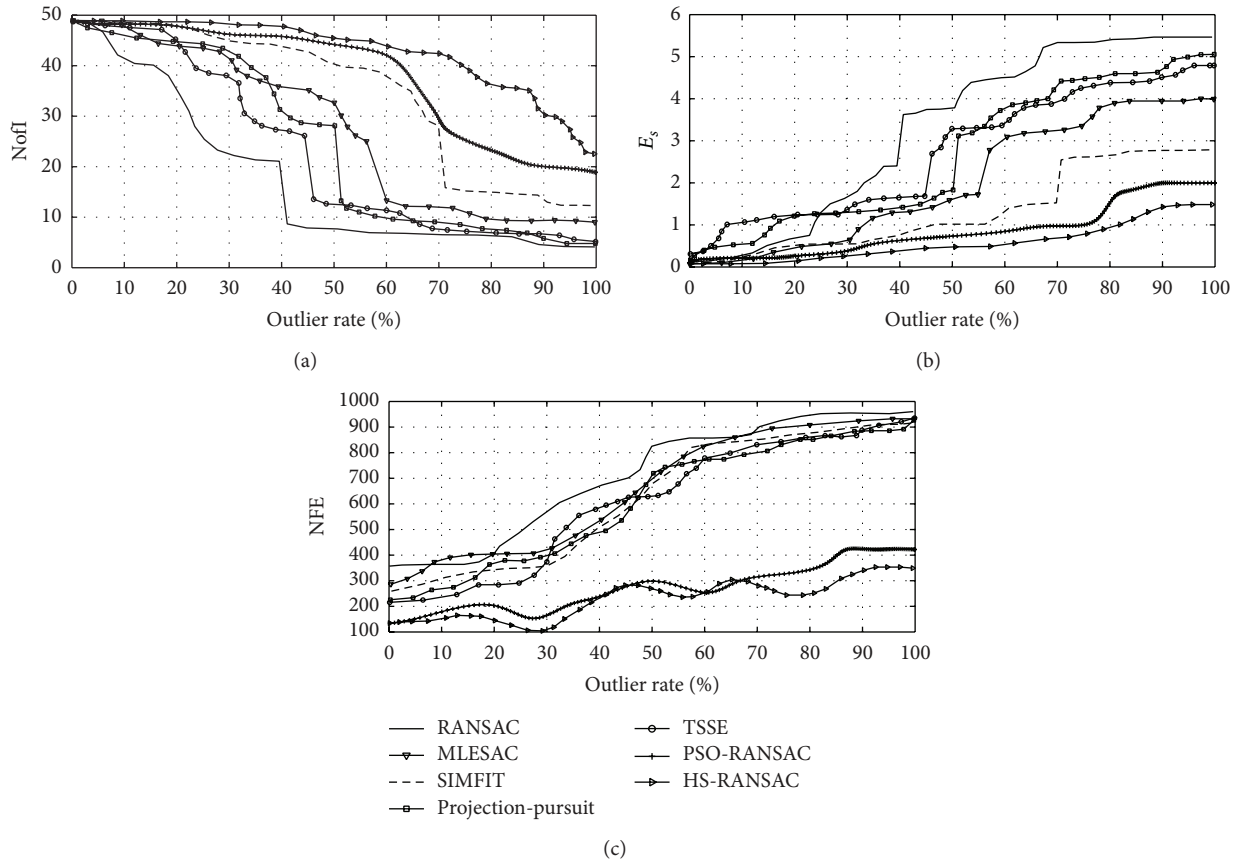


Figure 4: Experimental results corresponding to the estimation of **H** considering synthetic data.

five approaches are RANSAC-based estimators whose results are broadly known. In all cases, the algorithms are tuned according to the value set which is originally proposed by their own references. However, the PSO method has been included as a reference, only to validate the performance of the HS as an optimization approach.

In order to conduct a fair comparison between the HS version used in this work and PSO, an enhanced version of PSO has been also chosen with similar characteristics. Therefore, it is used in the comparisons, the PSO version reported in [54]. Such an approach is proposed to mitigate the premature convergence problem of the original PSO method. It incorporates two new elements: (1) a weight factor w and (2) a constriction factor $V_{\max}$. Similar to BW in the HS method, the weight factor w is linearly decreased during the algorithm execution to regulate the attraction force towards the best particle seen so far. On the other hand, the constriction factor $V_{\max}$ permits limiting the particle velocities in order to control their trajectories. Under such circumstances, the enhanced PSO version is used in combination with RANSAC considering the following configuration: $P = 10$, $c_1 = 2$, $c_2 = 2$, and Th $= 5$ whereas the weight factor w decreases linearly from 0.9 to 0.2. Additionally, the constriction factor $V_{\max}$ is fixed to 2. Such a configuration presents the best possible performance according to [54].

6.2.1. Homography Estimation with Synthetic Data. This section reports the experimental results corresponding to the estimation of homography matrix considering synthetic data. In the experiments, the same synthetic homography produced in Section 4.1 has been used (see Figure 3). The only difference is that the fraction of the incorporated outliers varies from 0 to 100%.

In the experiment, each algorithm's execution requires 1000 iterations. Since the proposed HS-RANSAC involves 50 initial evaluations (size of the harmony memory), it requires the execution of only 950 iterations to reach the 1000 evaluations. On the other hand, PSO-RANSAC possesses 10 particles; for this reason 100 generations need to be evolved in order to fulfill the 1000 iterations.

Figure 4 presents the performance for each algorithm. The results present the averaged outcomes obtained throughout 50 different executions. In order to appropriately analyze these results, it is necessary to define the concept of a breakdown point [18]. The breakdown point is identified as the highest outlier ratio from which the algorithm degrades its capacity to find inliers. It can be seen from Figure 4(a) that standard RANSAC has a breakdown point at 40%, the MLESAC at 55%, the SIMFIT method at 70%, the projection-pursuit algorithm at 50%, the TSSE at 45%, and the PSO-RANSAC at 80%. In contrast to such methods, the proposed approach, HS-RANSAC, does not seem to have a prominent

Table 3: Inlier detection comparison in terms of the detection rate (DR), the error (E_r), and the number of function evaluations (NFE) for standard RANSAC [14], the MLESAC [20], the SIMFIT method [21], the projection-pursuit algorithm [22], the TSSE [23], the PSO algorithm (PSO-RANSAC) [54], and the proposed HS-RANSAC approach, considering the four test images shown in Figures 5, 6, 7, and 8.

Image	Method	Detected inliers (NofI)	Missing	False alarms	DR (%)	E_r	NFE
(A) Total number of inliers (86)	Standard RANSAC	41	45	21	47.7	4.75	876
	MLESAC	55	31	14	63.9	3.11	852
	SIMFIT	62	24	11	72.0	2.98	842
	Projection-pursuit	58	28	12	67.4	3.53	798
	TSSE	48	38	14	55.8	3.42	815
	PSO-RANSAC	75	11	8	87.2	1.68	491
	HS-RANSAC	82	4	5	95.3	0.88	396
(B) Total number of inliers (72)	Standard RANSAC	32	40	18	44.4	3.98	765
	MLESAC	40	32	14	55.5	3.43	825
	SIMFIT	58	14	8	80.5	2.87	891
	Projection-pursuit	47	25	12	65.2	3.12	759
	TSSE	43	29	16	59.7	3.47	786
	PSO-RANSAC	63	9	5	87.5	1.51	374
	HS-RANSAC	70	2	3	97.2	0.79	328
(C) Total number of inliers (56)	Standard RANSAC	24	32	15	42.8	2.96	689
	MLESAC	27	29	11	48.2	2.41	628
	SIMFIT	42	14	9	75.0	1.98	724
	Projection-pursuit	37	19	13	66.0	2.85	754
	TSSE	32	24	14	57.1	2.74	776
	PSO-RANSAC	48	8	9	85.7	0.94	349
	HS-RANSAC	53	3	5	94.6	0.25	272
(D) Total number of inliers (122)	Standard RANSAC	62	60	22	50.8	4.02	832
	MLESAC	77	45	18	63.1	3.41	924
	SIMFIT	90	32	13	73.7	2.86	845
	Projection-pursuit	75	47	19	61.4	3.52	914
	TSSE	76	46	21	62.2	3.73	887
	PSO-RANSAC	110	12	10	90.1	1.41	427
	HS-RANSAC	115	7	5	94.2	0.51	338

breakdown point, since its capacity to detect inliers smoothly degrades. It is also observed that the HS-RANSAC algorithm presents the best performance in terms of the number of inliers (NofI), as it is able to detect most of them. For the estimated **H**, the error E_s (Figure 4(b)) is fairly comparable for all methods until they reach their breakdown points. Nonetheless, the proposed algorithm performed better, being the only algorithm that consistently found the minimum error at all outlier ratios.

In terms of number of function evaluations (NFE), Figure 4(c) shows that the standard RANSAC, the MLESAC, the projection-pursuit algorithm, and the TSSE invest approximately the same number of iterations for reaching their best estimation of **H**. Since such methods use a random walking algorithm as a search strategy, the NFE significantly grows as the number of outliers increases. On the other hand, the PSO-RANSAC and the HS-RANSAC (that use an optimization algorithm as search strategy) maintain a considerably low NFE value with independence of the number of outliers.

From the experiment, it is evident that the use of an optimization approach can considerably reduce the NFE value. However, there is no optimization algorithm suitable to find a good enough estimation considering the high multimodality and complex characteristics of the estimation process which is produced by the elevated contamination of the dataset. Therefore, although the PSO-RANSAC finds its best estimated fundamental matrix **H** investing approximately the same number of evaluations as the HS-RANSAC, such estimated matrix represents only a suboptimal solution. This fact can be observed in Figure 4(b) where it is clear that the PSO-RANSAC algorithm presents higher E_s values in comparison to the HS-RANSAC approach. The reason for this problem points to those operators used by PSO for modifying the individual positions. In PSO, during their evolution, the position of each agent in the next iteration is updated yielding an attraction towards the position of the best particle seen so far. Such behavior shows that the entire population, as the algorithm evolves, concentrates around the best particle, favoring the premature convergence (reaching suboptimal solutions) [53].

6.2.2. Homography Estimation with Real Images. In this section, the experimental results of the estimation of homographies **H** considering real images are reported. To evaluate the estimation performance of the proposed method, Table 3 tabulates the comparative inlier detection performance of the standard RANSAC [14], the MLESAC [20], the SIMFIT

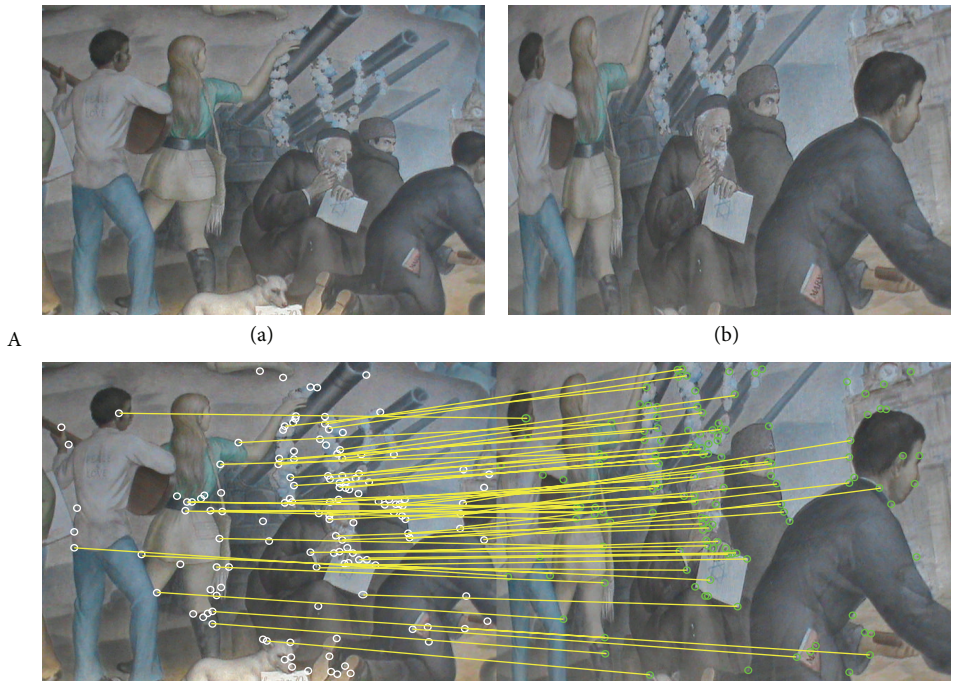


Figure 5: Test image "A": (a) first view, (b) second view, and (c) correspondence points and inliers produced by HS-RANSAC.

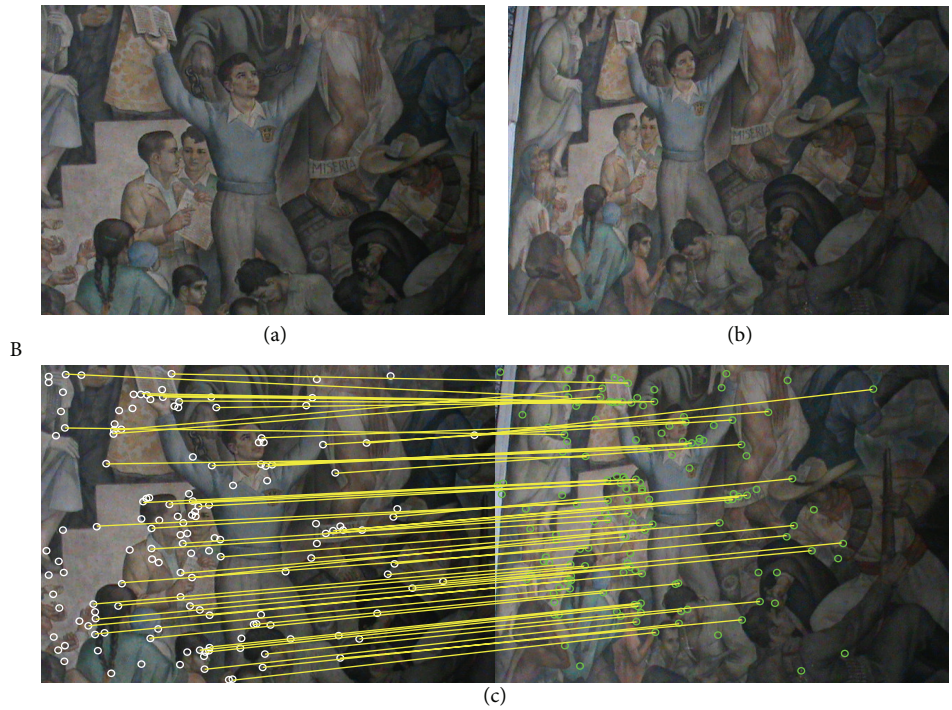


Figure 6: Test image "B": (a) first view, (b) second view, and (c) correspondence points and inliers produced by HS-RANSAC.

method [21], the projection-pursuit algorithm [22], the TSSE [23], the PSO algorithm (PSO-RANSAC) [54], and the proposed HS-RANSAC approach, in terms of the detection rate (DR), the error (E_r), and the number of function evaluations (NFE). The experimental dataset includes 4 images (images A, B, C, and D) which are shown in Figures 5, 6, 7, and 8. Such images contain a determined number of inliers which have been detected and counted by a human expert (A = 86, B = 72, C = 56, and D = 122). Such values act as ground truth for all the experiments. For the comparison, the detection rate

Figure 7: Test image "C": (a) first view, (b) second view, and (c) correspondence points and inliers produced by HS-RANSAC.

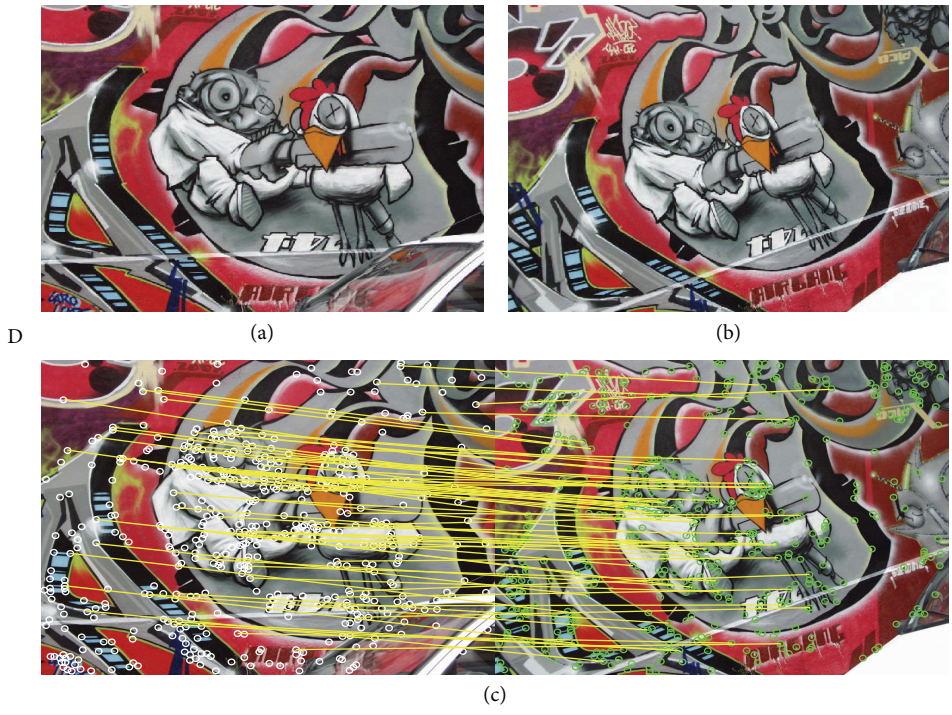


Figure 8: Test image "D": (a) first view, (b) second view, and (c) correspondence points and inliers produced by HS-RANSAC.

(DR) is defined as the ratio between the number of inliers correctly detected by the algorithm (NofI value) and the total number of inliers determined by the expert. The results consider 50 different executions for each algorithm over the four images. Experimental results show that the proposed HS method accomplishes at least a 94.2% of inlier detection accuracy. A close inspection of Table 3 also reveals that the proposed approach is able to achieve the smallest error (E_r), yet requiring a few number of function evaluations (NFE) for most cases.

Figures 5, 6, 7, and 8 also exhibit the results after applying the HS-RANSAC estimator. Such results present the median case obtained throughout 50 runs.

7. Engineering Application: Position Estimation in a Humanoid Robot

Additionally, in order to demonstrate the performance of the proposed approach in a real engineering application, the paper also reports the application of the HS-RANSAC to solve the problem of position estimation of a humanoid robot.

In the last decades, much work has already been accomplished in the area of humanoid robotics [56, 57]. Position determination for humanoid robots is a critical problem, since it is used to control their balance and locomotion. Recently, a notable research [58] has been devoted to achieving better performance in system position for humanoid robots by using sensor fusion methods. In general, integrating information from different sensors increases not only the versatility of the system, but also its cost and complexity. Vision is one of the most studied sensory modalities for position and navigation purposes since it provides rich information of the environment.

The framework of the approach presented in this section, as an application, is a vision system consisting of a fixed camera mounted on a Bioloid© humanoid robot. In the approach, the position (x, y) of the robot is computed considering the homography estimated by the HS-RANSAC. Therefore, the idea is to calculate the planar motion of the humanoid robot through the estimated homographies. Figure 9 illustrates the process of planar motion calculation.

The homography can be related to camera motion and plane location as follows:

$$\mathbf{H} = \mathbf{R} + \frac{1}{d}\mathbf{t}^T\mathbf{n}, \tag{11}$$

where d is the distance from the camera to the plane (the height of the humanoid approximately). $\mathbf{R}$ describes a rotation γ about the Z axis and can be expressed as

$$\mathbf{R} = \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{12}$$

And $\mathbf{t}$ is a translation vector with the form

$$\mathbf{t} = \left(t_x, t_y, 0\right). \tag{13}$$

As the unit normal $\mathbf{n}$ is $(0, 0, 1)$, considering the point $\mathbf{p}$, the rotation matrix $\mathbf{R}$, and the vector $\mathbf{t}$ (where $\mathbf{R}$ and $\mathbf{t}$ are calculated from the homography $\mathbf{H}$), the new planar position $\mathbf{p}_{\text{new}}$ can be computed as

$$\mathbf{p}_{\text{new}} = \mathbf{R}\mathbf{p} + \mathbf{t}. \tag{14}$$

More details about planar motion based on homography can be found in [59]. The HS-RANSAC algorithm and (11)–(14) were implemented in a Raspberry Pi. Since the computation must be verified in real time, the number of iterations is fixed to only 150. Figure 10 shows the calculated positions from the homographies estimated during the humanoid locomotion. Such a figure demonstrates that the information of the estimated position adequately reflects the humanoid movement in spite of the reduced number of iterations.

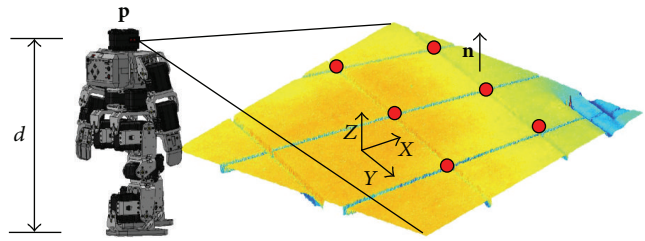


Figure 9: Process of planar motion calculation based on homographies.

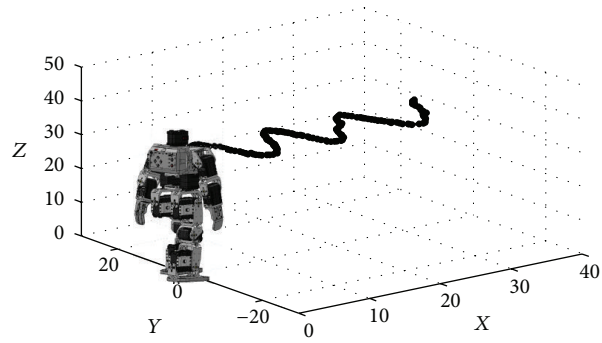


Figure 10: Position calculated from the homography.

8. Conclusions

In this paper, a new method for robustly estimating homographies from point correspondences based on the evolutionary algorithm has been presented. The approach combines the RANSAC method and the harmony search (HS) algorithm. With the combination, the proposed method adopts an alternative sampling strategy in comparison with RANSAC to build putative solutions. Under the new mechanism, new candidate solutions are generated iteratively by taking into consideration the quality of models produced by previous candidate solutions, instead of relying over a pure random selection as it is the case of RANSAC. On the other hand, a more accurate objective function was incorporated to adequately asses the quality of a candidate model. As a result, the proposed approach can substantially reduce the number of iterations still preserving the robust capabilities of RANSAC.

The proposed approach has been compared to other similar techniques proposed in the literature such as standard RANSAC [14], the MLESAC [17], the SIMFIT method [18], the projection-pursuit algorithm [19], the TSSE [20], and the PSO algorithm (PSO-RANSAC) [52]. The efficiency of the algorithm has been evaluated in terms of the detection rate (DR, NofI), accuracy (E_s, E_r), and computational cost

(NFE). Experimental results that consider real and synthetic data provide evidence on the remarkable performance of the proposed algorithm in comparison to such methods. Additionally, in order to demonstrate the performance of the proposed approach in a real engineering application, it has been employed to solve the problem of position estimation in a humanoid robot.

Although the experimental results indicate that the proposed method can yield better results on estimating homographies, it should be noticed that the aim of our paper is not intended to beat all the RANSAC methods which have been proposed earlier but to show that the use of evolutionary approaches can effectively serve as an attractive alternative to solve complex optimization problems, yet demanding fewer function evaluations.

Conflict of Interests

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

Acknowledgment

The present research has been supported under the Grant CONACYT CB 181053. The second author acknowledges the SEP of Mexico for its support (2014-2015).

References

[1] R. Szeliski and H.-Y. Shum, "Creating full view panoramic image mosaics and environment maps," in *Proceedings of the Conference on Computer Graphics (SIGGRAPH '97)*, pp. 251–258, August 1997.

[2] Y. He and R. Chung, "Image mosaicking for polyhedral scene and in particular singly visible surfaces," *Pattern Recognition*, vol. 41, no. 3, pp. 1200–1213, 2008.

[3] M. Brown and D. G. Lowe, "Automatic panoramic image stitching using invariant features," *International Journal of Computer Vision*, vol. 74, no. 1, pp. 59–73, 2007.

[4] A. Akyol and M. Gökmen, "Super-resolution reconstruction of faces by enhanced global models of shape and texture," *Pattern Recognition*, vol. 45, no. 12, pp. 4103–4116, 2012.

[5] H. Huang, H. He, X. Fan, and J. Zhang, "Super-resolution of human face image using canonical correlation analysis," *Pattern Recognition*, vol. 43, no. 7, pp. 2532–2543, 2010.

[6] K. Jia and S. Gong, "Hallucinating multiple occluded face images of different resolutions," *Pattern Recognition Letters*, vol. 27, no. 15, pp. 1768–1775, 2006.

[7] O. Deniz, G. Bueno, E. Bermejo, and R. Sukthankar, "Fast and accurate global motion compensation," *Pattern Recognition*, vol. 44, no. 12, pp. 2887–2901, 2011.

[8] E. Montijano and C. Sagues, "Distributed multi-camera visual mapping using topological maps of planar regions," *Pattern Recognition*, vol. 44, no. 7, pp. 1528–1539, 2011.

[9] J. Su, R. Chung, and L. Jin, "Homography-based partitioning of curved surface for stereo correspondence establishment," *Pattern Recognition Letters*, vol. 28, no. 12, pp. 1459–1471, 2007.

[10] T. T. Santos and C. H. Morimoto, "Multiple camera people detection and tracking using support integration," *Pattern Recognition Letters*, vol. 32, no. 1, pp. 47–55, 2011.

[11] H. Zhang and J. P. Ostrowski, "Visual motion planning for mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 2, pp. 199–208, 2002.

[12] C. Sagüés and J. J. Guerrero, "Visual correction for mobile robot homing," *Robotics and Autonomous Systems*, vol. 50, no. 1, pp. 41–49, 2005.

[13] G. López-Nicolás, J. J. Guerrero, and C. Sagüés, "Visual control of vehicles using two-view geometry," *Mechatronics*, vol. 20, no. 2, pp. 315–325, 2010.

[14] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the Association for Computing Machinery*, vol. 24, no. 6, pp. 381–395, 1981.

[15] E. İmre and A. Hilton, "Order statistics of RANSAC and their practical application," *International Journal of Computer Vision*, 2014.

[16] X. Wu, Q. Zhao, and W. Bu, "A SIFT-based contactless palmprint verification approach using iterative RANSAC and local palmprint descriptors," *Pattern Recognition*, vol. 47, no. 10, pp. 3314–3326, 2014.

[17] F. Zhou, Y. Cui, Y. Wang, L. Liu, and H. Gao, "Accurate and robust estimation of camera parameters using RANSAC," *Optics and Lasers in Engineering*, vol. 51, no. 3, pp. 197–212, 2013.

[18] J. Matas and O. Chum, "Randomized RANSAC with $T_{d,d}$ test," *Image and Vision Computing*, vol. 22, no. 10, pp. 837–842, 2004.

[19] C.-M. Cheng and S.-H. Lai, "A consensus sampling technique for fast and robust model fitting," *Pattern Recognition*, vol. 42, no. 7, pp. 1318–1329, 2009.

[20] P. H. S. Torr and A. Zisserman, "MLESAC: a new robust estimator with application to estimating image geometry," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, 2000.

[21] S. B. Heinrich, "Efficient and robust model fitting with unknown noise scale," *Image and Vision Computing*, vol. 31, no. 10, pp. 735–747, 2013.

[22] R. Subbarao and P. Meer, "Beyond RANSAC: user independent robust regression," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition Workshop (CVPRW '06)*, vol. no. 1, p. 101, June 2006.

[23] H. Wang and D. Suter, "Robust adaptive-scale parametric model estimation for computer vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1459–1474, 2004.

[24] L. Jun-Hua and L. Ming, "An analysis on convergence and convergence rate estimate of elitist genetic algorithms in noisy environments," *Optik*, vol. 124, no. 24, pp. 6780–6785, 2013.

[25] E. Mendel, R. A. Krohling, and M. Campos, "Swarm algorithms with chaotic jumps applied to noisy optimization problems," *Information Sciences*, vol. 181, no. 20, pp. 4494–4514, 2011.

[26] H. Pan, L. Wang, and B. Liu, "Particle swarm optimization for function optimization in noisy environment," *Applied Mathematics and Computation*, vol. 181, no. 2, pp. 908–919, 2006.

[27] H.-G. Beyer, "Evolutionary algorithms in noisy environments: theoretical issues and guidelines for practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2–4, pp. 239–267, 2000.

[28] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.

[29] M. Mahdavi, M. Fesanghary, and E. Damangir, "An improved harmony search algorithm for solving optimization problems,"

Applied Mathematics and Computation, vol. 188, no. 2, pp. 1567–1579, 2007.

[30] M. G. Omran and M. Mahdavi, "Global-best harmony search," *Applied Mathematics and Computation*, vol. 198, no. 2, pp. 643–656, 2008.

[31] K. S. Lee and Z. W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 194, no. 36–38, pp. 3902–3933, 2005.

[32] K. S. Lee, Z. W. Geem, S.-H. Lee, and K.-W. Bae, "The harmony search heuristic algorithm for discrete structural optimization," *Engineering Optimization*, vol. 37, no. 7, pp. 663–684, 2005.

[33] K. S. Lee and Z. W. Geem, "A new structural optimization method based on the harmony search algorithm," *Computers and Structures*, vol. 82, no. 9-10, pp. 781–798, 2004.

[34] J. H. Kim, Z. W. Geem, and E. S. Kim, "Parameter estimation of the nonlinear Muskingum model using harmony search," *Journal of the American Water Resources Association*, vol. 37, no. 5, pp. 1131–1138, 2001.

[35] Z. W. Geem, "Optimal cost design of water distribution networks using harmony search," *Engineering Optimization*, vol. 38, no. 3, pp. 259–280, 2006.

[36] Z. W. Geem, K. S. Lee, and Y. Park, "Application of harmony search to vehicle routing," *American Journal of Applied Sciences*, vol. 2, no. 12, pp. 1552–1557, 2005.

[37] D. Oliva, E. Cuevas, G. Pajares, D. Zaldivar, and M. Perez-Cisneros, "Multilevel thresholding segmentation based on harmony search optimization," *Journal of Applied Mathematics*, vol. 2013, Article ID 575414, 24 pages, 2013.

[38] E. Cuevas, N. Ortega-Sánchez, D. Zaldivar, and M. Pérez-Cisneros, "Circle detection by harmony search optimization," *Journal of Intelligent & Robotic Systems: Theory and Applications*, vol. 66, no. 3, pp. 359–376, 2012.

[39] A. Kaveh and S. Talatahari, "Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures," *Computers and Structures*, vol. 87, no. 5-6, pp. 267–283, 2009.

[40] S. Mun and Z. W. Geem, "Determination of individual sound power levels of noise sources using a harmony search algorithm," *International Journal of Industrial Ergonomics*, vol. 39, no. 2, pp. 366–370, 2009.

[41] S. Mun and Z. W. Geem, "Determination of viscoelastic and damage properties of hot mix asphalt concrete using a harmony search algorithm," *Mechanics of Materials*, vol. 41, no. 3, pp. 339–353, 2009.

[42] Q.-K. Pan, P. N. Suganthan, J. J. Liang, and M. F. Tasgetiren, "A local-best harmony search algorithm with dynamic sub-harmony memories for lot-streaming flow shop scheduling problem," *Expert Systems with Applications*, vol. 38, no. 4, pp. 3252–3259, 2011.

[43] Q.-K. Pan, P. N. Suganthan, M. F. Tasgetiren, and J. J. Liang, "A self-adaptive global best harmony search algorithm for continuous optimization problems," *Applied Mathematics and Computation*, vol. 216, no. 3, pp. 830–848, 2010.

[44] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "SURF: speeded up robust features," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[45] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the 4th Alvey Vision Conference*, pp. 147–151, 1988.

[46] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2nd edition, 2004.

[47] P. Meer, "Robust techniques in computer vision," in *Emerging Topics in Computer Vision*, G. Medioni and S. B. Kang, Eds., pp. 107–190, Prentice Hall, Boston, Mass, USA, 2004.

[48] E. Valian, S. Tavakoli, and S. Mohanna, "An intelligent global harmony search approach to continuous optimization problems," *Applied Mathematics and Computation*, vol. 232, pp. 670–684, 2014.

[49] V. Kumar, J. K. Chhabra, and D. Kumar, "Parameter adaptive harmony search algorithm for unimodal and multimodal optimization problems," *Journal of Computational Science*, vol. 5, no. 2, pp. 144–155, 2014.

[50] S. M. Ashrafi and A. B. Dariane, "Performance evaluation of an improved harmony search algorithm for numerical optimization: Melody Search (MS)," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 4, pp. 1301–1321, 2013.

[51] C. Hamzaçebi, "Improving genetic algorithms' performance by local search for continuous function optimization," *Applied Mathematics and Computation*, vol. 196, no. 1, pp. 309–317, 2008.

[52] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, December 1995.

[53] B. O. Arani, P. Mirzabeygi, and M. S. Panahi, "An improved PSO algorithm with a territorial diversity-preserving scheme and enhanced exploration-exploitation balance," *Swarm and Evolutionary Computation*, vol. 11, pp. 1–15, 2013.

[54] M. Clerc and J. Kennedy, "The particle swarm—explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[55] J. Chen, Q.-k. Pan, and J.-q. Li, "Harmony search algorithm with dynamic control parameters," *Applied Mathematics and Computation*, vol. 219, no. 2, pp. 592–604, 2012.

[56] A.-L. Lee and J.-H. Kim, "3-Dimensional pose sensor algorithm for humanoid robot," *Control Engineering Practice*, vol. 18, no. 10, pp. 1173–1182, 2010.

[57] H. Shahbazi, K. Jamshidi, A. H. Monadjemi, and H. Eslami, "Biologically inspired layered learning in humanoid robots," *Knowledge-Based Systems*, vol. 57, pp. 8–27, 2014.

[58] J. F. Seara and G. Schmidt, "Intelligent gaze control for vision-guided humanoid walking: methodological aspects," *Robotics and Autonomous Systems*, vol. 48, no. 4, pp. 231–248, 2004.

[59] B. Liang and N. Pears, "Visual navigation using planar homographies," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '02)*, vol. 1, pp. 205–210, 2002.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 149702, 9 pages
http://dx.doi.org/10.1155/2015/149702

Research Article

Application of Boosting Regression Trees to Preliminary Cost Estimation in Building Construction Projects

Yoonseok Shin

Department of Plant and Architectural Engineering, Kyonggi University, Gwanggyosan-ro 154-42, Yeongtong-gu, Suwon, Gyeonggi-do 443-760, Republic of Korea

Correspondence should be addressed to Yoonseok Shin; shinys@kgu.ac.kr

Received 8 October 2014; Revised 11 December 2014; Accepted 7 January 2015

Academic Editor: Rahib H. Abiyev

Among the recent data mining techniques available, the boosting approach has attracted a great deal of attention because of its effective learning algorithm and strong boundaries in terms of its generalization performance. However, the boosting approach has yet to be used in regression problems within the construction domain, including cost estimations, but has been actively utilized in other domains. Therefore, a boosting regression tree (BRT) is applied to cost estimations at the early stage of a construction project to examine the applicability of the boosting approach to a regression problem within the construction domain. To evaluate the performance of the BRT model, its performance was compared with that of a neural network (NN) model, which has been proven to have a high performance in cost estimation domains. The BRT model has shown results similar to those of NN model using 234 actual cost datasets of a building construction project. In addition, the BRT model can provide additional information such as the importance plot and structure model, which can support estimators in comprehending the decision making process. Consequently, the boosting approach has potential applicability in preliminary cost estimations in a building construction project.

1. Introduction

In building construction, budgeting, planning, and monitoring for compliance with the client's available budget, time, and work outstanding are important [1]. The accuracy of the construction cost estimation during the planning stage of a project is a crucial factor in helping the client and contractor with the adequate decision making and for the successful completion of the project [2–5]. However, there is a problem in that it is difficult to quickly and accurately estimate the construction costs at the early stage because the drawings and documentation are generally incomplete [6]. Machine learning approaches can be applied to alleviate this problem. Machine learning has some advantages over the human-crafted rules for data driven works, that is, accurate, automated, fast, customizable, and scalable [7].

Cost estimating approaches using a machine learning technique such as a neural network (NN) or support vector machine (SVM) have received significant attention since the early 1990s for accurately predicting the construction costs under a limited amount of project information. The NN model [1, 8–11] and the SVM model [12–16] were developed for predicting and/or estimating the construction costs. Although applying an NN to construction cost estimations has been very popular and has shown superior accuracy over other competing techniques [2, 4, 17–21], it has several disadvantages, such as a lack of self-learning and a time-consuming rule acquisition process [14]. A SVM, introduced by Vapnik [22], has attracted a great deal of attention because of its capacity for self-learning and high performance in generalization; moreover, it has shown the potential for utilization in construction cost estimations [5, 13, 14, 16, 23, 24]. However, the SVM approach requires a great deal of trial and error to determine a suitable kernel function [14]. Moreover, SVM models have a high level of algorithmic complexity and require extensive amounts of memory [25].

Among the recent machine learning techniques, the boosting approach, which was developed by Freund and Schapire [26], who also introduced the AdaBoost algorithm, has become an important application in machine learning and predicting models [27]. The boosting approach provides an effective learning algorithm and strong boundaries in

terms of the generalization performance [28–31]. Compared with competing techniques used for prediction problems, the performance of the boosting approach is superior to that of both a NN [32] and a SVM [33]. It is also simple, easy to program, and has few parameters to be tuned [31, 34, 35]. Because of these advantages, the boosting approach has been actively utilized in various domains. In the construction domain, some studies have attempted to apply this approach to the classification problem (for predicting a categorical dependent variable), such as the prediction of litigation results [27] and the selection of construction methods [31, 36]. However, there have been no efforts to do so for regression problems (for predicting a continuous dependent variable), such as construction cost estimation.

In this study, the boosting regression tree (BRT) is applied to the cost estimation at the early stage of a construction project to examine the applicability of the boosting approach for a regression problem within the construction domain. The BRT in this study is based on the module of a stochastic gradient boosting tree, which was proposed by Friedman (2002) [37]. It was developed as a novel advance in data mining that extends and improves the regression tree using a stochastic gradient boosting approach. Therefore, it has advantages of not only a boosting approach but also a regression tree, that is, high interpretability, conceptual simplicity, computational efficiency, and so on. The boosting approach can especially adopt the other data mining techniques, that is, a NN and SVM, as well as decision tree, as base learner. This feature matches up to the latest trends in the field of fusion of computational intelligence techniques to develop efficient computational models for solving practical problems.

In the next section, the construction cost estimation and its relevant studies are briefly reviewed. In the third section, the theory of a BRT and a cost estimation model using a BRT are both described. In the fourth section, the cost estimation model using a BRT is applied to a dataset from an actual project of a school building construction in Korea and is compared with that of an NN and an SVM. Finally, some concluding remarks and suggestions for further study are presented.

2. Review of Cost Estimation Literature

Raftery [38] categorized the preliminary cost estimation system used in building construction projects into three generations. The first generation of the system was a method from the late 1950s to the late 1960s that utilized the unit-price. The second generation of the system, which was developed from the middle of the 1970s, was a statistical method using a regression analysis according to propagating personal computers. The third generation of the system is a knowledge-based artificial intelligence method from the early 1980s. However, based on the third generation, Kim [39] also separated a fourth generation based on machine learning techniques such as a NN and SVM. The author showed an outstanding performance in construction cost estimation, although much remains to be resolved, for example, the complexity of the parameter settings.

We believe that the boosting approach can be a next-generation cost estimation system at the early stage of a construction project. In the prediction problem domain, combining the predictors of several models often results in a model with improved performance. The boosting approach is one such method that has shown great promise. Empirical studies have shown that combining models using the boosting approach produces a more accurate regression model [40]. In addition, the boosting approach can be extensively applied to prediction problems using an aforementioned machine learning technique such as a NN and SVM, as well as decision trees [27]. However, the boosting approach has never been used in regression problems of the construction domain, including cost estimations, but has been actively utilized in other domains, such as remote aboveground biomass retrieval [41], air pollution prediction [42], software effort estimation [43], soil bulk density prediction [44], and Sirex noctilio prediction [45]. In this study, we examine the applicability of a BRT for estimating the costs in the construction domain.

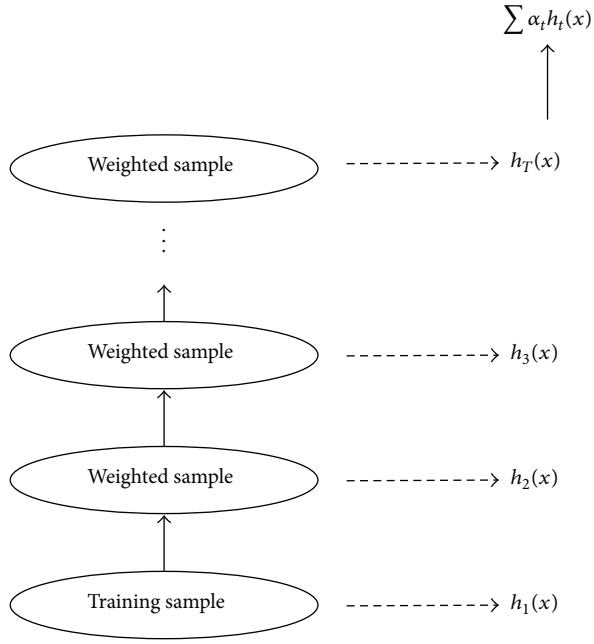


Figure 1: Schematic of a boosting procedure.

3. Boosting Regression Trees

Because of the abundance of exploratory tools, each having its own pros and cons, a difficult problem arises in selecting the best tool. Therefore, it would be beneficial to try to combine their strengths to create an even more powerful tool. To a certain extent, this idea has been implemented in a new family of regression algorithms referred to under the general term of "boosting." Boosting is an ensemble learning method for improving the predictive performance of a regression procedure, such as the use of a decision tree [46]. As shown in Figure 1, the method attempts to boost the accuracy of any given learning algorithm by fitting a series of models, each having a low error rate, and then combining them into

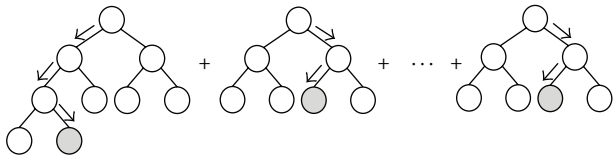


Figure 2: Gradient boosted decision tree ensemble.

an ensemble that may achieve better performance [36, 47]. This simple strategy can result in a dramatic improvement in performance and can be understood in terms of other well-known statistical approaches, such as additive models and a maximum likelihood [48].

Stochastic gradient boosting is a novel advance to the boosting approach proposed by Friedman [37] at Stanford University. Of the previous studies [26, 49–51] related to boosting for regression problems, only Breiman [50] alludes to involving the optimization of a regression loss function as part of the boosting algorithm. Friedman [52] proposed using the connection between boosting and optimization, that is, the gradient boost algorithm. Friedman [37] then showed that a simple subsampling trick can greatly improve the predictive performance of stochastic gradient boost algorithms while simultaneously reducing their computational time.

The stochastic gradient boost algorithm proposed by Friedman [37] uses regression trees as the basis functions. Thus, this boosting regression tree (BRT) involves generating a sequence of trees, each grown on the residuals of the previous tree [46]. Prediction is accomplished by weighting the ensemble outputs of all regression trees, as shown in Figure 2 [53]. Therefore, this BRT model inherits almost all of the advantages of tree-based models, while overcoming their primary disadvantages, that is, inaccuracies [54].

In these algorithms, the BRT approximates the function $F(x)$ as an additive expansion of the base learner (i.e., a small tree) [43]:

$$F(x) = F_0(x) + \beta_1 F_1(x) + \beta_2 F_2(x) + \cdots + \beta_m F_m(x). \quad (1)$$

A single base learner does not make sufficient prediction using the training data, even when the best training data are used. It can boost the prediction performance using a series of base learners with the lowest residuals.

Technically, BRT employs an iterative algorithm, where, at each iteration m, a new regression tree $h(x; \{R_{lm}\}_l^L)$ partitions the x-space into L-disjoint regions $\{R_{lm}\}_l^L$ and predicts a separate constant value in each one [54]:

$$h\left(x; \{R_{lm}\}_l^L\right) = \sum_{l-1}^{L} \overline{y}_{lm} \mid \left(x \in R_{lm}\right). \quad (2)$$

Here $\overline{y}_{lm} = \text{mean}_{x_i \in R_{lm}}(\widetilde{y}_{im})$ is the mean of pseudo-residuals (3) in each region R_{lm} induced at the mth iteration [37, 54]:

$$\widetilde{y}_{im} = -\left[\frac{\partial \Psi\left(y_i, F\left(x_i\right)\right)}{\partial F\left(x_i\right)}\right]_{F(x)=F_{m-1}(x)}. \quad (3)$$

The current approximation $F_{m-1}(x)$ is then separately updated in each corresponding region [37, 54]:

$$F_m(x) = F_{m-1}(x) + \upsilon \cdot \gamma_{lm} \mid \left(x \in R_{lm}\right), \quad (4)$$

where

$$\gamma_{lm} = \arg\min_{\gamma} \sum_{x_i \in R_{lm}} \Psi\left(y_i, F_{m-1}\left(x_i\right) + \gamma\right). \quad (5)$$

The "shrinkage" parameter υ controls the learning rate of the procedure.

This leads to the following BRT algorithm for generalized boosting of regression trees [37].

(1) Initialize $F(x)$, $F_0(x) = \arg\min_{\Upsilon} \sum_{i-1}^{N} \Psi(y_i, \gamma)$.

(2) For $m = 1$ to M do

(3) Select a subset randomly from the full training dataset,

$$\{\pi(i)\}_l^N = \text{rand\_perm}\{i\}_l^N. \quad (6)$$

(4) Fit the base learner,

$$\widetilde{y}_{\pi(i)m} = -\left[\frac{\partial \Psi\left(y_{\pi(i)m,}, F\left(x_{(i)}\right)\right)}{\partial F\left(x\left(i\right)\right)}\right]_{F(x)=F_{m-1}(x)}, \qquad i = 1, \widetilde{N}. \quad (7)$$

(5) Compute the model update for the current iteration,

$$\{R_{lm}\}_I^L = L - \text{terminal node tree}\left(\{\widetilde{y}_{\pi(i)m}, x_{\pi(i)}\}_l^{\widetilde{N}}\right). \quad (8)$$

(6) Choose a gradient descent step size as,

$$\gamma_{lm} = \arg\min_{\gamma} \sum_{x_{(i)} \in R_{lm}} \Psi\left(y_{\pi(i)}, F_{m-1}\left(x_{\pi(i)}\right) + \gamma\right). \quad (9)$$

(7) Update the estimate of $F(x)$ as,

$$F_m(x) = F_{m-1}(x) + \upsilon \cdot \gamma_{lm} \mid \left(x \in R_{lm}\right). \quad (10)$$

(8) end For.

There are specific algorithms for several loss criteria including least squares: $\psi(y, F) = (y - F)^2$, least-absolute deviation: $\psi(y, F) = |y - F|$, and Huber-M: $\psi(y, F) = (y - F)^2 \mid (|y - F| \preccurlyeq \delta) + 2\delta(|y - F| - \delta/2) \mid (|y - F| > \delta)$ [37]. The BRT applied in this study adopts the least squares for loss criteria as shown in Figure 3.

4. Application

4.1. Determining Factors Affecting Construction Cost Estimation. In general, the estimation accuracy in a building project is correlated with the amount of project information available regarding the building size, location, number of stories, and so forth [55]. In this study, the factors used for estimating the construction costs are determined in two steps. First, a

Table 1: Factors in construction cost estimation.

Description	Min.	Max	Average	Remark
Input				
Budget		(1) BTL (2) National finance		Nominal
School levels		(1) Elementary (2) Middle (3) High		Nominal
Land acquisition		(1) Existing (2) Building lots (3) Green belts		Nominal
Class number	12	48	31	Numerical
Building area (m^2)	1,204	3,863	2,694	Numerical
Gross floor area (m^2)	4,925	12,710	9,656	Numerical
Storey	3	7	4.7	Numerical
Basement floor (storey)	0	2	0.5	Numerical
Floor Height (m)	3.3	3.6	3.5	Numerical
Output				
Total construction cost (thousand KRW)	4,334,369	14,344,867	8,288,008	Numerical

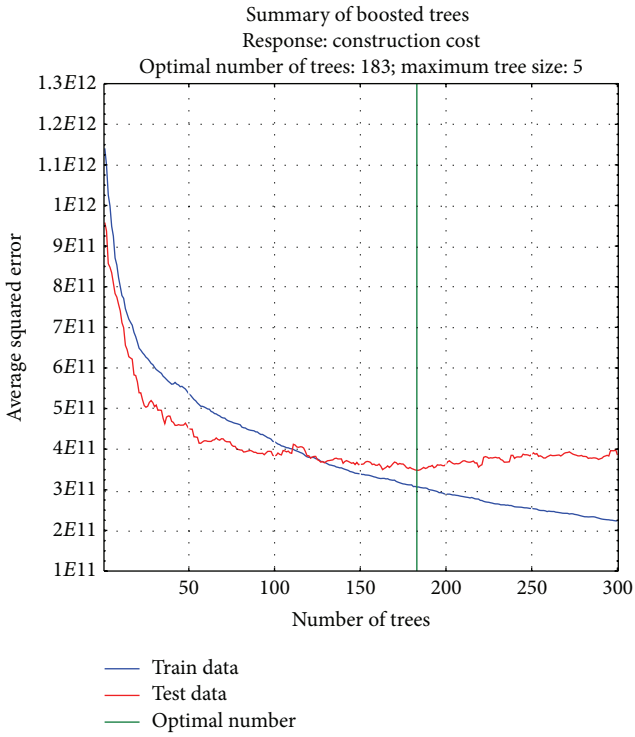


Figure 3: Training results of BRT.

list of factors affecting the preliminary cost estimation was made by reviewing previous studies [2, 3, 8, 12, 14, 20, 23, 55, 56]. Lastly, appropriate factors were selected from this list by interviewing practitioners who are highly experienced in construction cost estimation in Korea. Consequently, nine factors (i.e., input variables) were selected for this study, as shown in Table 1.

4.2. Data Collection. Data were collected from 234 completed school building projects executed by general contractors from 2004 to 2007 in Gyeonggi Province, Korea. These cost data were only the direct costs of different school buildings, such as elementary, middle, and high schools, without a markup as shown in Figure 4. According to the construction year, the total construction costs were converted using the Korean building cost index (BCI); that is, the collected cost data were multiplied by the BCI of the base year of 2005 (BCI = 1.00). The collected cost data of 217 school buildings were randomly divided into 30 test datasets and 204 training datasets.

4.3. Applying BRT to Construction Cost Estimation. In this study, the construction cost estimation model using a BRT was tested through application to real building construction projects. The construction costs were estimated using the BRT as follows. (1) The regression function $\widehat{F}(x)$ was trained using training data. In the dataset, the budget, school levels, gross floor area, and so on were allocated to each x_i of the training set. Each result, that is, the actual cost, was allocated to y_i. (2) After the training was completed according to the parameters such as the learning (shrinkage) rate, the number of additive trees, and the maximum and minimum number of levels, the series of trees $\widehat{F}(x)$ which maps x to y of training data set (y_i, x_i) with minimized loss function $\Psi(y_i, F(x_i))$ was found. (3) The expected value of $\widehat{F}(x)$, that is, the expected cost, was calculated for a new test dataset (y_j, x_j).

The construction cost estimation model proposed in this study was constructed using "STATISTICA Release 7." STATISTICA employs an implementation method usually

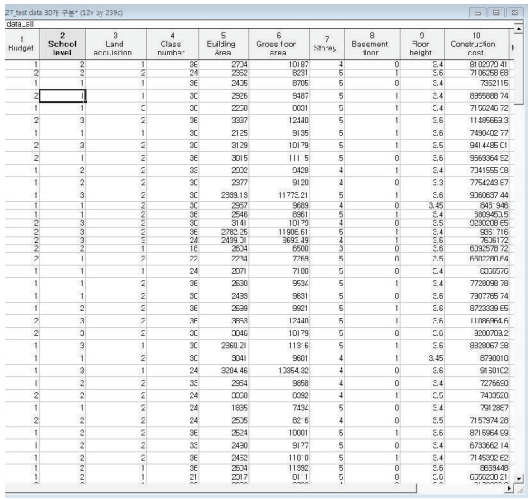


Figure 4: Fragment of cost dataset.

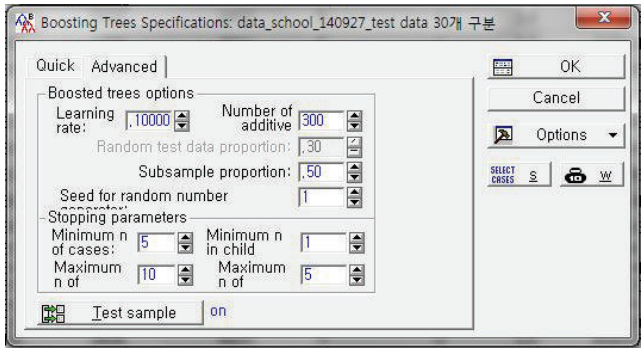


Figure 5: Parameter setting for BRT.

referred to as a stochastic gradient boosting tree by Friedman (2002, 2001) [37, 52], also known as TreeNet (Salford Systems, Inc.) or MART (Jerill, Inc.). In this software, a stochastic gradient booting tree is used for regression problems to predict a continuous dependent variable [57]. To operate a boosting procedure in STATISTICA, the parameter settings, that is, the learning rate, the number of additive trees, the proportion of subsampling, and so forth, are required. Firstly, the learning rate was set as 0.1. It was found that small values, that is, values under 0.1, lead to much better results in terms of the prediction error [52]. We empirically obtained the other parameters, which are shown in Figure 5. As a result, the training result of the BRT showed that the optimal number of additive trees is 183 and the maximum size of tree is 5, as shown in Figure 3.

4.4. Performance Evaluation. In general, the cost estimation performance can be measured based on the relationship between the estimated and actual costs [56]. In this study, the performance was measured using the Mean Absolute Error Rates (MAERs), which were calculated using

$$\text{MAERs} = \frac{\left(\sum \left|\left(\left(C_e - C_a\right)/C_a\right) \times 100\right|\right)}{n}, \quad (11)$$

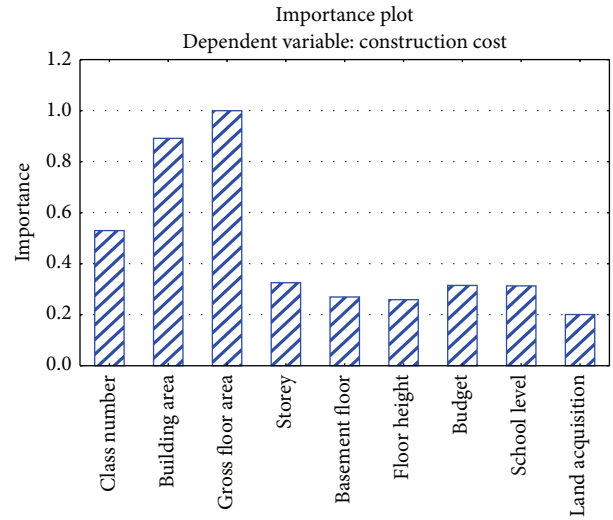


Figure 6: Importance plot of dependent variables.

where C_e is the estimated construction costs by model application, C_a is the actual construction costs collected, and n is the number of test datasets.

To verify the performance of the BRT model, the same cases were applied to a model based on a NN and the results compared. We chose the NN model because it showed a superior performance in terms of cost estimation accuracy in previous studies [2, 5, 14]. "STATISTICA Release 7" was also used to construct the NN model in this study. To construct a model using a NN, the optimal parameters have to be selected beforehand, that is, the number of hidden neurons, the momentum, and the learning rate for the NN. Herein, we determined the values from repeated experiments.

5. Results and Discussion

5.1. Results of Evaluation. The results from the 30 test datasets using a BRT and a NN are summarized in Tables 2 and 3. The results from the BRT model had MAERs of 5.80 with 20% of the estimates within 2.5% of the actual error rate, while 80% were within 10%. The NN model had MAERs of 6.05 with 10% of the estimates within 2.5% of the actual error rate, while 93.3% were within 10%. In addition, the standard deviations of the NN and BRT models are 3.192 and 3.980, respectively, as shown in Table 4.

The MAERs of two results were then compared using a t-test analysis. The MAERs of the two results are statistically similar, although there are differences between them. As the null hypothesis, the MAERs of the two results are all equal ($H_0 : u_D = 0$). The t-value is 0.263 and the P value is 0.793 (>0.05). Thus, the null hypothesis is accepted. This analysis shows that the MAERs of the two results are statistically similar.

The BRT model provided comprehensible information regarding the new cases to be predicted, which is an advantage inherent to a decision tree. Initially, the importance of each dependent variable to cost estimation was provided, as shown in Figure 6. These values indicate the importance

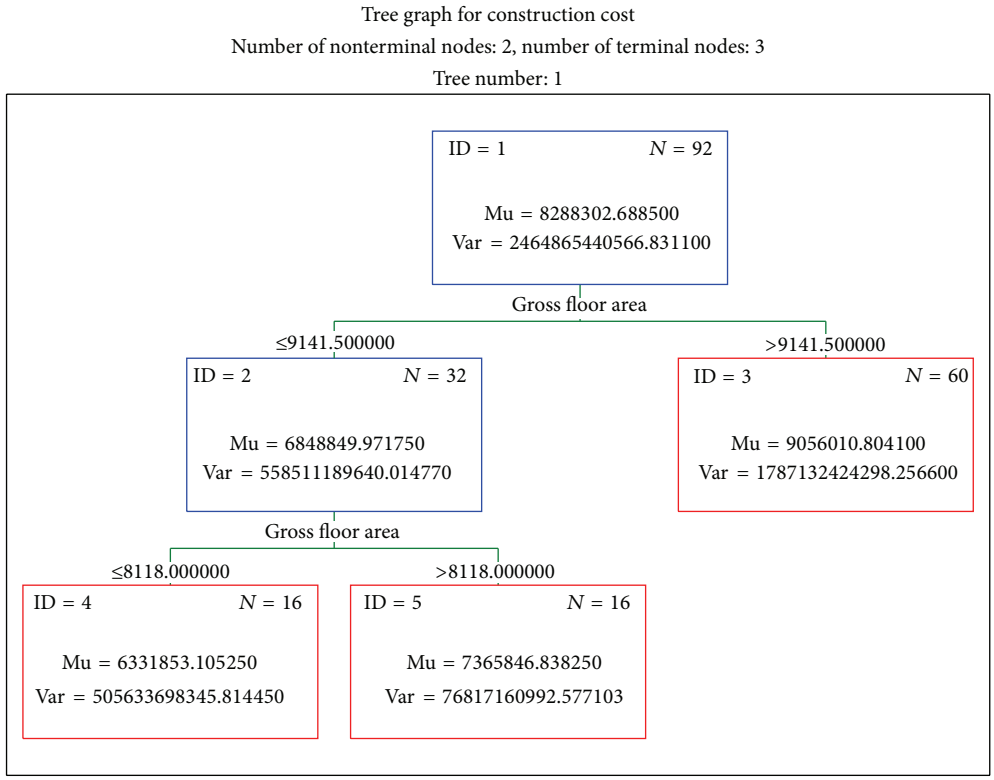


Figure 7: An example of structure model.

Table 2: Summary of results by estimation model.

Error rate (%)	NN		BRT	
	Fre. (%)	Cum. (%)	Fre. (%)	Cum. (%)
0.0–2.5	3 (10.0)	3 (10.0)	6 (20.0)	6 (20.0)
2.5–5.0	11 (36.7)	14 (46.7)	10 (33.3)	16 (53.3)
5.0–7.5	6 (20.0)	20 (66.7)	6 (20.0)	22 (73.3)
7.5–10.0	8 (26.7)	28 (93.3)	2 (6.7)	24 (80.0)
10.0–12.5	1 (3.3)	29 (96.7)	3 (10.0)	27 (90.0)
12.5–15.0	1 (3.3)	30 (100)	2 (6.7)	29 (96.7)
15.0–17.5	0 (0)	30 (100)	1 (3.3)	30 (100)
MAERs	6.05	—	5.80	—

of each variable for the construction cost estimation in the model. Finally, the tree structures in the model were provided as shown in Figure 7. This shows the estimation rules, such as the applied variables and their influence on the proposed model. Thus, an intuitive understanding of the whole structure of the model is possible.

5.2. Discussion of Results. This study was conducted using 234 school building construction projects. In addition, 30 of these projects were used for testing. In terms of the estimation accuracy, the BRT model showed slightly better results than the NN model, with MAERs of 5.80 and 6.05, respectively. In terms of the construction cost estimation, it is difficult to conclude that the performance of the BRT model is superior to that of the NN model because the gap between the two is not statistically different. However, even the similar performance of the BRT model is notable because the NN model has proven its superior performance in terms of cost estimation accuracy in previous studies. Similarly, in predicting the software project effort, Elish [43] compared the estimation accuracy of neural network, linear regression, support vector regression (SVR), and BRT. Consequently, BRT outperformed the other techniques in terms of the estimation performance that has been also achieved by SVR. These results mean that the BRT has remarkable performance in regression problem as well as classification one. Moreover, the BRT model provided additional information, that is, an importance plot and structure model, which helps the estimator comprehend the decision making process intuitively.

Consequently, these results reveal that a BRT, which is a new AI approach in the field of construction, has potential applicability in preliminary cost estimations. It can assist estimators in avoiding serious errors in predicting the construction costs when only limited information is available during the early stages of a building construction project. Moreover, a BRT has a large utilization possibility because the boosting approach can employ existing AI techniques such as a NN and SVM, along with decision trees, as base learners during the boosting procedure.

Table 3: Cost estimation results of each test set.

Number	Historical cost (1,000 KRW)	Neural networks		Boosting regression tree	
		Predicted cost (1,000 KRW)	Error rate (%)	Predicted cost (1,000 KRW)	Error rate (%)
1	6,809,450	7,704,034	13.14	7,206,795	5.84
2	9,351,716	10,015,906	7.10	9,805,656	4.85
3	6,656,230	7,251,317	8.94	6,322,112	5.02
4	7,119,470	7,128,513	0.13	7,418,373	4.20
5	7,304,747	7,978,990	9.23	7,349,178	0.61
6	9,729,392	9,516,946	2.18	9,259,162	4.83
7	10,801,826	9,817,999	9.11	9,682,119	10.37
8	7,944,318	7,246,763	8.78	7,136,773	10.17
9	10,879,004	10,136,431	6.83	10,572,777	2.81
10	7,552,814	7,764,300	2.80	7,683,295	1.73
11	8,845,099	8,558,536	3.24	8,370,497	5.37
12	10,690,800	10,001,503	6.45	10,015,284	6.32
13	8,694,721	8,258,452	5.02	8,446,796	2.85
14	6,582,636	6,810,406	3.46	6,954,507	5.65
15	7,583,680	8,312,216	9.61	8,194,292	8.05
16	7,099,220	7,955,966	12.07	8,292,381	16.81
17	8,145,147	8,604,444	5.64	8,522,009	4.63
18	8,652,810	7,853,765	9.23	8,270,169	4.42
19	10,527,278	10,040,039	4.63	9,611,194	8.70
20	6,679,924	6,467,344	3.18	7,397,923	10.75
21	8,383,830	9,203,887	9.78	8,487,286	1.23
22	7,298,932	8,018,225	9.85	8,294,895	13.65
23	7,505,428	7,749,053	3.25	7,967,265	6.15
24	7,710,921	7,622,053	1.15	7,795,563	1.10
25	6,196,652	6,503,022	4.94	5,940,634	4.13
26	8,897,861	8,554,455	3.86	8,714,123	2.06
27	7,840,787	8,535,617	8.86	8,863,975	13.05
28	8,023,067	7,666,898	4.44	6,900,068	14.00
29	7,495,213	7,270,806	2.99	7,695,613	2.67
30	7,653,005	8,003,292	4.58	7,775,139	1.60
MAERs			6.05		5.80

Table 4: Descriptive analysis of error rate estimation.

	MAERs	Std, deviation	Std, error	95% confidence interval of the MAERs	
				Lower	Upper
NN	6.045	3.192	0.583	2.542	4.291
BRT	5.800	3.980	0.727	3.170	5.351

6. Conclusion

This study applied a BRT to construction cost estimation, that is, the regression problem, to examine the applicability of the boosting approach to a regression problem in the construction domain. To evaluate the performance of the BRT model, its performance was compared with that of an NN model, which had previously proven its high performance capability in the cost estimation domains. The BRT model showed similar results when using 234 actual cost datasets of a building construction project in Korea. Moreover, the BRT model can provide additional information regarding the variables to support estimators in comprehending the decision making process. These results demonstrated that the BRT has dual advantages of boosting and decision trees. The boosting approach has great potential to be a leading technique in next generation construction cost estimation systems.

In this study, an examination using a relatively small dataset and number of variables was carried out on the performance of a BRT for construction cost estimation. Although

both models performed satisfactorily, further detailed experiments and analyses regarding the quality of the collected data are necessary to utilize the proposed model for an actual project.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was supported by Kyonggi University Research Grant 2012.

References

[1] G.-H. Kim, J.-E. Yoon, S.-H. An, H.-H. Cho, and K.-I. Kang, "Neural network model incorporating a genetic algorithm in estimating construction costs," *Building and Environment*, vol. 39, no. 11, pp. 1333–1340, 2004.

[2] G. H. Kim, S. H. An, and K. I. Kang, "Comparison of construction cost estimating models based on regression analysis, neural networks, and case-based reasoning," *Building and Environment*, vol. 39, no. 10, pp. 1235–1242, 2004.

[3] G. H. Kim and S. H. An, "A study on the correlation between selection methods of input variables and number of data in estimating accuracy: cost estimating using neural networks in apartment housing projects," *Journal of the Architectural Institute of Korea*, vol. 23, no. 4, pp. 129–137, 2007.

[4] H.-G. Cho, K.-G. Kim, J.-Y. Kim, and G.-H. Kim, "A comparison of construction cost estimation using multiple regression analysis and neural network in elementary school project," *Journal of the Korea Institute of Building Construction*, vol. 13, no. 1, pp. 66–74, 2013.

[5] G. H. Kim, J. M. Shin, S. Kim, and Y. Shin, "Comparison of school building construction costs estimation methods using regression analysis, neural network, and support vector machine," *Journal of Building Construction and Planning Research*, vol. 1, no. 1, pp. 1–7, 2013.

[6] S. H. An and K. I. Kang, "A study on predicting construction cost of apartment housing using experts' knowledge at the early stage of projects," *Journal of the Architectural Institute of Korea*, vol. 21, no. 6, pp. 81–88, 2005.

[7] H. Brink, *Real-World Machine Learning*, Manning, 2014.

[8] R. A. McKim, "Neural network applications to cost engineering," *Cost Engineering*, vol. 35, no. 7, pp. 31–35, 1993.

[9] I.-C. Yeh, "Quantity estimating of building with logarithm-neuron networks," *Journal of Construction Engineering and Management*, vol. 124, no. 5, pp. 374–380, 1998.

[10] J. Bode, "Neural networks for cost estimation: simulations and pilot application," *International Journal of Production Research*, vol. 38, no. 6, pp. 1231–1254, 2000.

[11] S. K. Kim and I. W. Koo, "A neural network cost model for office buildings," *Journal of the Architectural Institute of Korea*, vol. 16, no. 9, pp. 59–67, 2000.

[12] M.-Y. Cheng and Y.-W. Wu, "Construction conceptual cost estimates using support vector machine," in *Proceedings of the 22nd International Symposium on Automation and Robotics in Construction (ISARC '05)*, Ferrara, Italy, September 2005.

[13] U. Y. Park and G. H. Kim, "A study on predicting construction cost of apartment housing projects based on support Vector regression at the early project stage," *Journal of the Architectural Institute of Korea*, vol. 23, no. 4, pp. 165–172, 2007.

[14] S. H. An, K. I. Kang, M. Y. Cho, and H. H. Cho, "Application of support vector machines in assessing conceptual cost estimates," *Journal of Computing in Civil Engineering*, vol. 21, no. 4, pp. 259–264, 2007.

[15] F. Kong, X. Wu, and L. Cai, "Application of RS-SVM in construction project cost forecasting," in *Proceedings of the 4th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM '08)*, Dalian, China, October 2008.

[16] J.-M. Shin and G.-H. Kim, "A study on predicting construction cost of educational building project at early stage using support vector machine technique," *The Journal of Educational Environment Research*, vol. 11, no. 3, pp. 46–54, 2012.

[17] J. M. de la Garza and K. G. Rouhana, "Neural networks versus parameter-based applications in cost estimating," *Cost Engineering*, vol. 37, no. 2, pp. 14–18, 1995.

[18] R. Creese and L. Li, "Cost estimation of timber bridge using neural networks," *Cost Engineering*, vol. 37, no. 5, pp. 17–22, 1995.

[19] H. Adeli and M. Wu, "Regularization neural network for construction cost estimation," *Journal of Construction Engineering and Management*, vol. 124, no. 1, pp. 18–24, 1998.

[20] T. M. S. Elhag and A. H. Boussabaine, "An artificial neural system for cost estimation of construction projects," in *Proceedings of the 14th ARCOM Annual Conference*, September 1998.

[21] M. W. Emsley, D. J. Lowe, A. R. Duff, A. Harding, and A. Hickson, "Data modelling and the application of a neural network approach to the prediction of total construction costs," *Construction Management and Economics*, vol. 20, no. 6, pp. 465–472, 2002.

[22] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, London, UK, 1999.

[23] M. Hongwei, "An improved support vector machine based on rough set for construction cost prediction," in *Proceedings of the International Forum on Computer Science-Technology and Applications (IFCSTA '09)*, December 2009.

[24] M. Y. Cheng, H. S. Peng, Y. W. Wu, and T. L. Chen, "Estimate at completion for construction projects using evolutionary support vector machine inference model," *Automation in Construction*, vol. 19, no. 5, pp. 619–629, 2010.

[25] P. R. Kumar and V. Ravi, "Bankruptcy prediction in banks and firms via statistical and intelligent techniques: a review," *European Journal of Operational Research*, vol. 180, no. 1, pp. 1–28, 2007.

[26] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, part 2, pp. 119–139, 1997.

[27] D. Arditi and T. Pulket, "Predicting the outcome of construction litigation using boosted decision trees," *Journal of Computing in Civil Engineering*, vol. 19, no. 4, pp. 387–393, 2005.

[28] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 130–136, San Juan, Puerto Rico, USA, June 1997.

[29] C. P. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," in *Proceedings of the IEEE*

6th International Conference on Computer Vision, pp. 555–562, January 1998.

[30] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, "Boosting the margin: a new explanation for the effectiveness of voting methods," *The Annals of Statistics*, vol. 26, no. 5, pp. 1651–1686, 1998.

[31] Y. Shin, D. W. Kim, J. Y. Kim, K. I. Kang, M. Y. Cho, and H. H. Cho, "Application of adaboost to the retaining wall method selection in construction," *Journal of Computing in Civil Engineering*, vol. 23, no. 3, pp. 188–192, 2009.

[32] E. Alfaro, N. García, M. Gámez, and D. Elizondo, "Bankruptcy forecasting: an empirical comparison of AdaBoost and neural networks," *Decision Support Systems*, vol. 45, no. 1, pp. 110–122, 2008.

[33] E. A. Park, *A comparison of SVM and boosting methods and their application for credit scoring [M.S. thesis]*, Seoul National University, 2005.

[34] Y. Freund and R. E. Schapire, "A short introduction to boosting," *Journal of Japanese Society for Artificial Intelligence*, vol. 14, no. 5, pp. 771–780, 1999.

[35] Y.-S. Lee, H.-J. Oh, and M.-K. Kim, "An empirical comparison of bagging, boosting and support vector machine classifiers in data mining," *Korean Journal of Applied Statistics*, vol. 18, no. 2, pp. 343–354, 2005.

[36] Y. Shin, T. Kim, H. Cho, and K. I. Kang, "A formwork method selection model based on boosted decision trees in tall building construction," *Automation in Construction*, vol. 23, pp. 47–54, 2012.

[37] J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, 2002.

[38] J. Raftery, "The state of cost/modelling in the UK construction industry: a multi criteria approach," in *Building Cost Modeling and Computers*, P. S. Brandon, Ed., pp. 49–71, E&PN Spon, London, UK, 1987.

[39] G. H. Kim, *Construction cost prediction system based on artificial intelligence at the project planning stage [Ph.D. thesis]*, Korea University, Seoul, Republic of Korea, 2004.

[40] G. Ridgeway, "Generallized boosted model: A guide to the gbm package," CiteSeerx, 2005, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.151.4024.

[41] A. M. Filippi, İ. Güneralp, and J. Randall, "Hyperspectral remote sensing of aboveground biomass on a river meander bend using multivariate adaptive regression splines and stochastic gradient boosting," *Remote Sensing Letters*, vol. 5, no. 5, pp. 432–441, 2014.

[42] D. C. Carslaw and P. J. Taylor, "Analysis of air pollution data at a mixed source location using boosted regression trees," *Atmospheric Environment*, vol. 43, no. 22-23, pp. 3563–3570, 2009.

[43] M. O. Elish, "Improved estimation of software project effort using multiple additive regression trees," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10774–10778, 2009.

[44] M. P. Martin, D. L. Seen, L. Boulonne et al., "Optimizing pedotransfer functions for estimating soil bulk density using boosted regression trees," *Soil Science Society of America Journal*, vol. 73, no. 2, pp. 485–493, 2009.

[45] R. Ismail and O. Mutanga, "A comparison of regression tree ensembles: predicting Sirex noctilio induced water stress in Pinus patula forests of KwaZulu-Natal, South Africa," *International Journal of Applied Earth Observation and Geoinformation*, vol. 12, no. 1, pp. S45–S51, 2010.

[46] T. J. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, New York, NY, USA, 2nd edition, 2009.

[47] R. E. Schapire, "A brief introduction to boositng," in *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI '99)*, vol. 2, pp. 1401–1406, Stockholm, Sweden, July-August 1999.

[48] J. Friedman, T. Hastie, and R. Tibshirani, "Additive statistical regression: a statistical view of boosting," *The Annals of Statistics*, vol. 28, pp. 337–407, 2000.

[49] H. Drucker, "Improving regressors using boosting techniques," in *Proceedings of the 14th International Conference on Machine Learning*, Nashville, Tenn, USA, July 1997.

[50] L. Breiman, "Prediction games and arcing algorithms," *Neural Computation*, vol. 11, no. 7, pp. 1493–1517, 1999.

[51] G. Ridgeway, D. Madigan, and T. Richardson, "Boosting methodology for regression problems," in *Proceedings of the 7th International Workshop on Artificial Intelligence and Statistics*, Fort Lauderdale, Fla, USA, January 1999.

[52] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.

[53] J. Ye, J.-H. Chow, J. Chen, and Z. Zheng, "Stochastic gradient boosted distributed decision trees," in *Proceedings of the ACM 18th International Conference on Information and Knowledge Management (CIKM '09)*, pp. 2061–2064, Hong Kong, November 2009.

[54] J. H. Friedman and J. J. Meulman, "Multiple additive regression trees with application in epidemiology," *Statistics in Medicine*, vol. 22, no. 9, pp. 1365–1381, 2003.

[55] M. Skitmore, "The effect of project information on the accuracy of building price forecasts," in *Building Cost Modeling and Computers*, P. S. Brandon, Ed., E & FN SPON, London, UK, 1987.

[56] M. Skitmore, "Early stage construction price forecasting: a review of performance," Occasional Paper, Royal Institute of Chartered Surveyors, London, UK, 1991.

[57] T. Hill and P. Lewicki, *STATISTICS: Methods and Applications*, StatSoft, Tulsa, Okla, USA, 1st edition, 2006.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 921487, 6 pages
http://dx.doi.org/10.1155/2015/921487

Research Article

A Simple Fitness Function for Minimum Attribute Reduction

Yuebin Su,[1,2] Jin Guo,[1] and Zejun Li[2]

[1]*School of Information Science and Technology, Southwest Jiaotong University, Chengdu 610031, China*
[2]*School of Science, Sichuan University of Science and Engineering, Zigong 643000, China*

Correspondence should be addressed to Yuebin Su; mathsyb@163.com

Received 18 August 2014; Revised 29 November 2014; Accepted 18 December 2014

Academic Editor: Rahib H. Abiyev

The goal of minimal attribute reduction is to find the minimal subset R of the condition attribute set C such that R has the same classification quality as C. This problem is well known to be NP-hard. When only one minimal attribute reduction is required, it was transformed into a nonlinearly constrained combinatorial optimization problem over a Boolean space and some heuristic search approaches were used. In this case, the fitness function is one of the keys of this problem. It required that the fitness function must satisfy the equivalence between the optimal solution and the minimal attribute reduction. Unfortunately, the existing fitness functions either do not meet the equivalence, or are too complicated. In this paper, a simple and better fitness function based on positive domain was given. Theoretical proof shows that the optimal solution is equivalent to minimal attribute reduction. Experimental results show that the proposed fitness function is better than the existing fitness function for each algorithm in test.

1. Introduction

For a given dataset, attribute reduction is a fundamental problem in rough set theory as proposed by Pawlak and Sowinski [1]. Formally, it is a nonlinearly constrained combinatorial optimization problem whose objective function is the size of a candidate subset of attributes and whose constraints, represented in terms of classification quality of attribute subsets, are the conditions met by an attribute reduction. As shown by [2], this problem has been proven to be NP-hard.

To overcome this problem, many strategies had to be considered in the literatures during the past two decades. In general, there are two kinds of categories for minimum attribute reduction: greedy (or hill-climbing) categories and stochastic categories. The greedy categories usually employ rough set attribute significance as heuristic knowledge. It starts off with an empty set or attribute core and then adopts forward selection or backward elimination. Hu and Cercone give a reduction algorithm using the positive region-based attribute significance as the guiding heuristic [3]. Wang et al. develop a conditional information entropy-based reduction algorithm, using conditional entropy-based attribute significance [4]. Hu et al. compute the significance of an attribute making use of heuristic ideas from discernibility matrices and propose a heuristic reduction algorithm [5]. Susmaga considers both indiscernibility and discernibility relations in attribute reduction [6]. These categories are fast but do not guarantee to find an optimal or minimal reduction.

Some researchers use stochastic methods for rough set attribute reduction. These categories are optimization methods. It has a higher probability of finding a minimum reduct than the first category. Wroblewski combines a genetic algorithm with a greedy algorithm to generate short reducts. However, it uses highly time-consuming operations and cannot assure that the resulting subset is really a reduct [7]. Taking Wroblewski's work as a foundation, Bjorvand and Komorowski apply genetic algorithms to compute approximate reducts [8]. The algorithm makes several variations and practical improvements both in speed and in the quality of approximation. The reduct generation algorithms based on genetic algorithms for the rough set attribute reduction are quite efficient [9].

But rough set can only deal with the discrete attributes. A method for discretization based on particle swarm optimization (PSO) is presented in [10]. Taking this work as a foundation, an algorithm for knowledge reduction in rough sets is proposed based on particle swarm optimization in [11]. This algorithm can solve some problems that the existing heuristic

algorithm cannot solve. In order to improve the efficiency of the algorithm, many scholars constantly improve and update these algorithms. Santana-Quintero Luis et al. present a multiobjective evolutionary algorithm which consists of a hybrid between a particle swarm optimization approach and some concepts from rough sets theory [12]. The main idea of the approach is to combine the high convergence rate of the particle swarm optimization algorithm with a local search approach based on rough sets which is able to spread the nondominated solutions found. After that, two-step particle swarm optimization to solve the feature selection problem was given by Bello et al. in [13]. The improved algorithm is a method which can improve the search efficiency. Chi et al. presented a method for continuous attribute discretization based on quantum PSO algorithm [14]. Hsieh and Horng presented a method for feature selection based on asynchronous discrete PSO search algorithm [15]. Also, other stochastic algorithms were used to attribute reduction, for example, ant colony algorithm (ACO) [16] and support vector machine (SVR) [17].

For systems where the optimal or minimal subset is required, stochastic category may be used. In this case, this problem is transformed into a problem of finding a maximum (or minimum) value of a fitness function at first, and then some stochastic optimization method is applied to solve the fitness maximization (or minimization) problem. A common way to transform a constrained optimization problem into an unconstrained fitness optimization problem is to use penalty methods [18]. For such methods, designing a better fitness function is the most important work. To get good performance, the fitness functions should meet the requirements that the fitness evaluation of a candidate solution is appropriate and the optimality equivalence is guaranteed. Here, the optimality equivalence means that the optimal solution of the fitness maximization problem corresponds to a minimum attribute reduction. Unfortunately, the existing fitness functions do not well meet the above mentioned requirements and consequently affect the performance of the related algorithms [19].

In this paper, an applicable fitness function was proposed. Compared with the existing fitness functions as mentioned earlier, it not only takes into account less factors but also overcomes the drawback. The experimental results show that, for each of the two tested algorithms, the use of the proposed fitness function has a higher probability to find a minimum reduction than the use of the function proposed in [19].

The rest of the paper is organized as follows. Section 2 presents some concepts about minimum attribute reduction and reviewed and analysed a fitness function proposed in [19]. In Section 3, a new fitness function and properties are presented. In Section 4, the results of experiments and comparison analysis are given. Finally, Section 5 concludes the paper.

2. Basic Notions and Related Works

In this section, we will review some basic notions in the theory of rough sets which are necessary for the description of the minimum attribute reduction problem.

A decision table can be represented as $S = \{U, A, V, f\}$, where $U = \{x_1, x_2, \ldots, x_n\}$ is a nonempty finite set of objects, $A = C \cup D$, where C is a set of condition attributes and D is a decision attribute set, V is the domain of attributes belonging to A, and $f : U \times A \to V$ is a function assigning attribute values to objects in U.

With any $R \subseteq A$, there is an associated indiscernibility relation IND(R):

$$\mathrm{IND}(R) = \{(x, y)\, f(x, a) = f(y, a),\ \forall a \in R, x, y \in U\}. \tag{1}$$

Let $X \subseteq U$; the R-lower approximation of X is defined as $\underline{R}X = \{x \in U \mid [x]_R \subseteq X\}$, where $[x]_R$ denotes an equivalence class of IND(R) determined by object x. The notation $\mathrm{POS}_R(D)$ refers to the R-positive region given by $\mathrm{POS}_R(D) = \cup_{X \in U/\mathrm{IND}(D)} \underline{R}X$. The R-approximation quality with respect to decisions attribute set D is defined as $\gamma_R = |\mathrm{POS}_R(D)|/|U|$.

For decision table $S = \{U, A, V, f\}$, it may have many attribute reductions; the set of reductions is defined as

$$\begin{aligned}\mathrm{Red} = \{&R \subseteq C \mid \mathrm{POS}_R(D) = \mathrm{POS}_C(D),\\ &\forall B \subset R, \mathrm{POS}_R(D) \neq \mathrm{POS}_B(D)\}.\end{aligned} \tag{2}$$

For attribute reduction, the minimal attribute reduction with minimal cardinality will be searched. The minimal attribute reduction problem can be formulated as the following nonlinearly combinational optimization problem:

$$\begin{aligned}&\min_{R \subseteq C} \quad |R|\\ &\text{s.t.} \quad \begin{cases}\mathrm{POS}_R(D) = \mathrm{POS}_C(D),\\ \mathrm{POS}_R(D) \neq \mathrm{POS}_B(D), \quad \forall B \subset R.\end{cases}\end{aligned} \tag{3}$$

Let Red_m be a set which was defined as follows:

$$\mathrm{Red}_m = \{R \in \mathrm{Red} \mid \forall B \in \mathrm{Red}, |B| \geq |R|\}. \tag{4}$$

By the definition of Red_m, the following proposition is apparent.

Proposition 1. *Let $R \subset C$; then R is the optimal solution of the optimization problem (3) if and only if $R \in Red_m$.*

According to Proposition 1, each element of Red_m corresponds to a minimal reduction. In order to solve problem (3), the most commonly used approach is to transform it into the following unconstrained maximization problem and to solve it by heuristic algorithms:

$$\max_{R \subseteq C} F(R), \tag{5}$$

where $F(R)$ is a fitness function about attribute subset R.

For this optimization problem, the equivalence of optimality between the minimum attribute reduction problem (3) and the fitness maximization of the function $F(R)$ must be guaranteed. Unfortunately, most of the functions do not satisfy the requirement in the literatures [19].

For instance, Ye et al. defined a fitness function in [19]. Let

$$\mathrm{IND}(C) = \left\{[x_1]_C, [x_2]_C, \ldots, [x_N]_C\right\}, \tag{6}$$

where the cardinality of the equivalence classes $[x_i]_C$ meets $|[x_i]_C| \le |[x_{i+1}]_C|$, where $|\cdot|$ denotes the cardinality of a set. The function was defined as

$$F_1(R) = \frac{m - |R|}{m} + \frac{n}{\Delta}\gamma_R, \tag{7}$$

where $\Delta = |[x_1]_C| > 0$.

Let $R \subset C$, $|R| = |C| - 1$, and $|\mathrm{POS}_R(D)| = |\mathrm{POS}_C(D)| - 1$; then $F_1(C) = |\mathrm{POS}_C(D)|/\Delta$ and $F_1(R) = 1/m + (|\mathrm{POS}_C(D)| - 1)/\Delta$, and thus $F_1(C) - F_1(R) = 1/\Delta - 1/m$. If $1/\Delta = 1/m$, then $F_1(C) - F_1(R) = 0$. It suggests that the fitness function cannot distinguish between the two. That is to say, the two are equivalent when running the algorithm. This drawback will reduce the algorithmic performance. Moreover, in practical applications, since contains classification quality γ_R, the value of the function $F_1(R)$ is not an integer. In the search process, the calculation error will reduce the probability of searching a minimum reduction. At the end, the consistency of decision table must be determined before calculating the fitness value, which increases the computational complexity.

3. A New Function and Its Properties

In this section, we present a new fitness function. Firstly, the relevant definitions were introduced as follows.

Definition 2. Let $R \subseteq C$. If $\mathrm{POS}_R(D) = \mathrm{POS}_C(D)$ holds, then R is said to be a consistent set of C. Let CS be a set which includes all the consistent sets of C [19]:

$$CS = \left\{R \mid \mathrm{POS}_R(D) = \mathrm{POS}_C(D), R \subseteq C\right\}. \tag{8}$$

By the definition of reduction, for any consistent set $R \in CS$, R either is a reduction itself or contains a reduction. If R is minimal, then R is a reduction of C. It is very obvious that $\mathrm{Red} \subseteq CS$. A nonlinearly constrained combinatorial optimization problem correlated with CS is defined as

$$\min_{R \in CS} |R|. \tag{9}$$

Let CS_m be a set which is defined as follows:

$$CS_m = \{R \in CS : \forall B \in CS, |B| \ge |R|\}. \tag{10}$$

By the definition of CS_m, we know that CS_m is a subset of CS and the following proposition is apparent.

Proposition 3. *$\forall R \in CS_m$, then R is a reduction.*

Proposition 4. *Let $R \subset C$; then $R \in CS_m$ if and only if R is the optimal solution of the optimization problem (9).*

By Proposition 4, CS_m is the solution set of the optimization problem (9). According to the previous definition, we present a new fitness function as follows: $\forall R \subseteq C$,

$$F(R) = m\left(\left|\mathrm{POS}_C(D)\right| - \left|\mathrm{POS}_R(D)\right|\right) + |R|, \tag{11}$$

where $m = |C|$.

Proposition 5. *Let $P \subset C$.*

(1) If $P \in CS, \forall R \in CS_{\min}$, then $F(R) \le F(P) \le m$.

(2) If $P \notin CS$, then $F(P) > m$.

In particular, $F(P) > F(B), \forall B \in CS$.

Proof. If $P \in CS$, $\forall R \in CS_{\min}$, then $|\mathrm{POS}_P(D)| = |\mathrm{POS}_R(D)| = |\mathrm{POS}_C(D)|$ and $|P| \le |R| \le m$. That is $F(R) \le F(P) \le m$ according to the definition of the fitness function $F(R)$. The (1) is hold. If $P \notin CS$, then $|\mathrm{POS}_C(D)| > |\mathrm{POS}_P(D)|$, thus $F(P) = m(|\mathrm{POS}_C(D)| - |\mathrm{POS}_P(D)|) + |P| > m + |P| > m$. □

According to Proposition 5, the codomain of the function $F(R)$ can be divided into two disjoint sets $[\min F(R), m]$ and $(m, \max F(R))$, where $\min F(R)$ and $\max F(R)$ are the minimum and maximum value of $F(R)$, respectively. By (2), the defect of the function proposed in [19] was avoided. A minimum optimization problem is defined as follows by the definition of $F(R)$:

$$\min_{R \subset C} F(R). \tag{12}$$

Theorem 6. *Let $R \subset C$; then $R \in CS_m$ if and only if R is a minimum attribute reduction; that is, $CS_m = \mathrm{Red}_m$.*

Proof. $R \in CS_m$; then R is a reduction by Proposition 3. If R is not a minimum attribute reduction, then there is a minimum attribute reduction B such that $|B| < |R|$.

On the other hand, since B is an attribute reduction, then $B \in CS$. By the definition of CS_m and $R \in CS_m$, $|B| \ge |R|$, a contradiction and vice versa. □

By Theorem 6, all the elements of CS_m are minimum attribute reduction; then according to Proposition 4, we can get that all the solutions of the optimization problem (9) are the minimum attribute reductions.

Theorem 7. *$\forall R \subset CS_m$, then R is an optimal solution of the optimization problem (12). It is equivalent to the case where all the elements of CS_m are the optimal solutions of the optimization problem (12).*

Proof. We use proof by contradiction. Assume that there is a $R \in CS_m$ such that R is not an optimal solution of the optimization problem (12). Then $\exists B \subseteq C$ such that $F(B) < F(R)$. So

$$\begin{aligned} 0 &> F(B) - F(R) \\ &= m\left(\left|\mathrm{POS}_R(D)\right| - \left|\mathrm{POS}_B(D)\right|\right) + |B| - |R|. \end{aligned} \tag{13}$$

Then, there are two cases of B that will be discussed in the following.

(1) If $B \in CS$, then $\mathrm{POS}_B(D) = \mathrm{POS}_R(D) = \mathrm{POS}_C(D)$, so $|\mathrm{POS}_B(D)| - |\mathrm{POS}_R(D)| = 0$ and $F(B) - F(R) = |R| - |B|$. Since $R \in CS_m$, thus $|R| - |B| \le 0$. Consequently

$$0 > F(B) - F(R) = |B| - |R| \ge 0, \tag{14}$$

a contradiction.

Table 1: Parameter settings for both algorithms.

Algorithms	p-size	$V_{\max}$	$I_{\max}$	c_1	c_2	m_p	c_p
PSO	20	7	100	2.0	2.0	—	—
Gen	50	—	100	—	—	0.4	0.6

(2) If $B \notin CS$, then $|\mathrm{POS}_B(D)| < |\mathrm{POS}_R(D)| = |\mathrm{POS}_C(D)|$; that is, $|\mathrm{POS}_R(D)| - |\mathrm{POS}_B(D)| \geq 1$, and then

$$0 > F(B) - F(R) \geq m + |B| - |R| \geq |B| \geq 1, \quad (15)$$

a contradiction. □

From the above proof we know that all the elements of CS_m are the optimal solution of the optimization problem (12); that is to say, the minimum attribute reduction is the optimal solution of the optimization problem (12).

Theorem 8. *Let $R \subset C$; if R is an optimal solution of the optimization problem (12), then $R \in CS_m$.*

Proof. (1) $R \in CS$.

If $R \notin CS$, then $|\mathrm{POS}_R(D)| < |\mathrm{POS}_C(D)|$. For all $B \in CS$, $\mathrm{POS}_B(D) = \mathrm{POS}_C(D)$. Since R is an optimal solution of the optimization problem (12), then $0 \geq F(R) - F(B)$, so

$$\begin{aligned} 0 &\geq F(R) - F(B) \\ &= m\left(\left|\mathrm{POS}_B(D)\right| - \left|\mathrm{POS}_R(D)\right|\right) + |R| - |B| \\ &\geq m + |R| - |B| \geq 1, \end{aligned} \quad (16)$$

a contradiction, so $R \in CS$.

(2) $R \in CS_m$.

If $R \notin CS_m$, let $B \in CS_m$. According to (1) we know $R \in CS$, so $\mathrm{POS}_B(D) = \mathrm{POS}_R(D)$, and then

$$0 \geq F(R) - F(B) = |R| - |B| > 0, \quad (17)$$

a contradiction, so $R \in CS_m$. □

According to Theorem 8, the optimal solution of the optimization problem (12) is a minimum attribute reduction. Then, by Theorems 7 and 8, we can obtain the following theorem obviously.

Theorem 9. *Let $R \subset C$; then R is an optimal solution of the optimization problem (12) if and only if $R \in CS_m$.*

By Theorem 9, we can get that the optimal solution of the optimization problem (12) is equivalent to the minimal attribute reduction. Then, according to Theorem 6, we can obtain the following theorem.

Theorem 10. *Let $R \subset C$; then R is a minimum attribute reduction if and only if R is an optimal solution of the optimization problem (12).*

By Theorem 10, the minimum value of the function proposed in this paper is equivalent to the minimal attribute reduction. We may get the minimum attribute reduction by searching the minimum value of function, and then stochastic category can be used.

4. Performance Comparison

In order to analyze and evaluate the effectiveness of the fitness function proposed in this paper, in the literature [19], an experiment was designed in the following way. Two existing minimum attribute reduction algorithms based on different types of stochastic optimization technique are used in the comparison. The first is the particle swarm optimization-based attribute reduction algorithm proposed in [20], denoted by PSO, and the other is a genetic algorithm based attribute reduction algorithm presented in [7, 8, 16], denoted by GA in short. In experiments, both algorithms were implemented and tested on a number of datasets using two different fitness functions: one is proposed in [19] (F_1 in short) and the new fitness function is proposed in this paper (F in short). Seven datasets were chosen from the UCI machine learning repository. Most of these datasets are commonly used for evaluating attribute reduction algorithms in the literature [7, 8, 11, 14, 16, 17, 19].

For each of the two fitness functions, both algorithms were run 50 times on each of the datasets with the same setting of the parameters involved. For each run, three values need to be recorded: the first is the length of the output, the second is the output which corresponds to a reduction or not, and the last is run time. If an output is a reduction, then the output is said to be a normal output; otherwise, it is an unsuccessful output. If the length of the normal output is minimal, then the output is said to be a successful output. Let STL denote the length of the successful output, AVL denote the average length, and AVT denote the average run time during the 50 runs. The ratios of successful and normal outputs are denoted, respectively, by s_1 and s_2.

A PC running Windows 7 (32-bit) with 2.1 × 2 GHz CPU and 2 GB of main memory was used to run both algorithms. Both algorithms were programed by the MATLAB. Parameter settings for both algorithms were shown in Table 1. In Table 1, p-size refers to the size of population (in GA) or particle swarm (in PSO), $I_{\max}$ is the maximum allowed number of iterations (or generations), $V_{\max}$ is the upper bound on velocity needed in the PSO algorithm, m_p and c_p are the probabilities of mutation and crossover in GA, and c_1 and c_2 are the learning coefficients in PSO. Tables 2 and 3 present the main performance of PSO and GA using each of the two fitness functions.

From Table 2, for the algorithm PSO, on the index of STL, both fitness functions have the same value. It means that both fitness functions can output the minimum reduction. On the index of AVL, the value of F is not more than F_1's except for the last date. It reflects that the output of F is more focused on STL. The same conclusion can be arrived at from the index s_1. It means that it has a higher probability to find a minimum reduction by using the proposed fitness function. For AVT,

Table 2: Performance of PSO using different fitness functions.

Dataset	F_1				F			
	STL	AVL	s_1/s_2	AVT	STL	AVL	s_1/s_2	AVT
blance	4	4	1/1	35.392	4	4	1/1	26.467
spect	16	16	1/1	7.471	16	16	1/1	6.921
Lung-cancer	3	4.02	0.08/1	2.628	3	**3.92**	**0.22**/1	1.096
Sponge	8	8.4	0.61/0.98	5.327	8	**8.02**	**0.79**/1	4.217
Vote	8	8.2	0.8/1	7.419	8	**8.13**	**0.87**/1	5.522
Zoo	5	5.05	0.96/1	5.092	5	**5**	**1**/1	4.275
Soybean-small	2	**2.32**	**0.78**/1	2.091	2	2.38	0.72/1	1.162

Table 3: Performance of GA using different fitness functions.

Dataset	F_1				F			
	STL	AVL	s_1/s_2	AVT	STL	AVL	s_1/s_2	AVT
blance	4	4	1/1	38.501	4	4	1/1	30.359
spect	16	16	1/1	8.021	16	16	1/1	7.856
Lung-cancer	3	4.21	0.07/0.98	2.983	3	**4.03**	**0.2/1**	1.123
Sponge	8	8.58	0.59/0.96	6.306	8	**8.17**	**0.77**/0.98	4.311
Vote	8	8.31	0.78/1	8.762	8	**8.22**	**0.82**/1	5.762
Zoo	5	5.11	0.94/1	6.243	5	**5**	**1**/1	4.52
Soybean-small	2	**2.43**	**0.76**/1	2.227	2	2.51	0.71/1	1.175

all the values of F are less than the values of F_1. Therefore, the experimental data show that the efficiency of the proposed fitness function is better than F_1 by using the algorithm PSO. The same conclusion can be obtained from Table 3 by using the algorithm GA.

The above two experiments show that the proposed fitness function is more adequate than the other fitness functions on the datasets.

5. Conclusions

In this paper, in order to overcome the drawback of the existing fitness functions for the problem of minimum attribute reduction, we discussed the fitness function and a simpler fitness function was proposed in this paper. Theoretical analysis and experimental results show that it can ensure the optimality equivalence and is more adequate than the existing fitness function.

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 The Science and Technology Development Plan of The Railway Ministry of China (2012X003-A), Scientific Research Fund of Sichuan Provincial Education Department (14ZA0245, 14ZB0208), and the Opening Project of Sichuan Province University Key Laboratory of Bridge Non-Destruction Detecting and Engineering Computing (2014QYJ02). The authors are grateful to the anonymous reviewers for their valuable comments and suggestions.

References

[1] Z. Pawlak and R. Sowinski, "Rough set approach to multi-attribute decision analysis," *European Journal of Operational Research*, vol. 72, no. 3, pp. 443–459, 1994.

[2] S. K. Wong and W. Ziarko, "On optimal decision rules in decision tables," *Bulletin of the Polish Academy of Sciences*, vol. 33, no. 11-12, pp. 693–696, 1985.

[3] X. Hu and N. Cercone, "Learning in relational databases: a rough set approach," *Computational Intelligence*, vol. 11, no. 2, pp. 323–338, 1995.

[4] G. Y. Wang, H. Yu, and D. C. Yang, "Decision table reduction based on conditional information entropy," *Chinese Journal of Computers*, vol. 25, no. 7, pp. 759–766, 2002.

[5] K. Hu, Y. Lu, and C. Shi, "Feature ranking in rough sets," *AI Communications*, vol. 16, no. 1, pp. 41–50, 2003.

[6] R. Susmaga, "Reducts and constructs in attribute reduction," *Fundamenta Informaticae*, vol. 61, no. 2, pp. 159–181, 2004.

[7] J. Wroblewski, "Finding minimal reducts using genetic algorithms," in *Proceedings of the 2nd Annual Joint Conference on Information Sciences*, pp. 186–189, Wrightsville Beach, NC, USA, October 1995.

[8] A. T. Bjorvand and J. Komorowski, "Practical applications of genetic algorithms for efficient reduct computation," in *Proceedings of the 15th IMACS World Congress on Scientific Computation, Modelling and Applied Mathematics, Berlin, August 1997*, A. Sydow, Ed., vol. 4, pp. 601–606, Wissenschaft & Technik, 1997.

[9] A. Skowron, H. Wang, A. Wojna, and J. Bazan, "A hierarchical approach to multi-modal classification," in *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, D. Slezak, G. Wang, M. Szczuka, I. Duentsch, and Y. Y. Yao, Eds., vol. 3642 of

Lecture Notes in Computer Science, pp. 119–127, Springer, Berlin, Germany, 2005.

[10] F. M. Ma, X. D. Zhang, and L. Wang, "A new algorithm for discr etization in rough sets based on particle swarm optimization," in *Proceedings of the International Symposium on Intelligence Computation and Applications*, pp. 472–476, 2005.

[11] W. Dong, L. Xu, and J. Wang, "An application of DPSO algorithms to rough sets knowledge reduction," in *Proceedings of the 17th Chinese Control and Decision Conference*, pp. 1845–1847, 2005.

[12] V. Santana-Quintero Luis, N. Ramirez-Santiago, and A. Carlos, "A new proposal for multiobjective optimization us ing particle swarm optimization and rough sets theory," in *Proceedings of the 9th International Conference on Parallel Problem Solving from Nature (PPSN '06)*, vol. 4193, pp. 483–492, 2006.

[13] R. Bello, Y. Gomez, A. Nowe, and M. M. Garcia, "Two-step particle swarm optimization to solve the feature selection problem," in *Proceedings of the 7th International Conference on Intelligent Systems Design and Applications (ISDA '07)*, pp. 691–696, Rio de Janeiro, Brazil, 2007.

[14] Y. Chi, Y. Dong, K. Xia, and J. Shi, "Continuous attribute discretization based on quantum PSO algorithm," in *Proceedings of the 7th World Congress on Intelligent Control and Automation (WCICA '08)*, pp. 6187–6191, Chongqing, China, June 2008.

[15] W.-T. Hsieh and S.-J. Horng, "Feature selection based on asynchronous discrete particle swarm optimal search algorithm," in *Proceedings of the 5th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP '12)*, pp. 262–268, December 2012.

[16] R. Jensen and Q. Shen, "Finding rough set reducts with ant colony optimization," in *Proceedings of the 2003 UK Workshop on Computational Intelligence*, pp. 15–22, 2003.

[17] Y. Zhao, Y. Zhang, and N. Xiong, "Biological data classification using rough sets and support vector machines," in *Proceedings of the Annual Meeting of the North American Fuzzy Information Processing Society*, pp. 344–349, 2009.

[18] R. Fletcher, *Practical Methods of Optimization*, John Wiley & Sons, New York, NY, USA, 2nd edition, 1987.

[19] D. Ye, Z. Chen, and S. Ma, "A new fitness function for solving minimum attribute reduction problem," in *Rough Set and Knowledge Technology*, vol. 6401 of *Lecture Notes in Computer Science*, pp. 118–125, Springer, Berlin, Germany, 2010.

[20] X. Wang, J. Yang, X. Teng, W. Xia, and R. Jensen, "Feature selection based on rough sets and particle swarm optimization," *Pattern Recognition Letters*, vol. 28, no. 4, pp. 459–471, 2007.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 939606, 10 pages
http://dx.doi.org/10.1155/2015/939606

Research Article

An Intelligent Model for Pairs Trading Using Genetic Algorithms

Chien-Feng Huang,[1] Chi-Jen Hsu,[1] Chi-Chung Chen,[2] Bao Rong Chang,[1] and Chen-An Li[1]

[1]*Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan*
[2]*Department of Electrical Engineering, National Chiayi University, Chiayi City 60004, Taiwan*

Correspondence should be addressed to Bao Rong Chang; brchang@nuk.edu.tw

Received 21 December 2014; Revised 6 March 2015; Accepted 14 March 2015

Academic Editor: Rahib H. Abiyev

Pairs trading is an important and challenging research area in computational finance, in which pairs of stocks are bought and sold in pair combinations for arbitrage opportunities. Traditional methods that solve this set of problems mostly rely on statistical methods such as regression. In contrast to the statistical approaches, recent advances in computational intelligence (CI) are leading to promising opportunities for solving problems in the financial applications more effectively. In this paper, we present a novel methodology for pairs trading using genetic algorithms (GA). Our results showed that the GA-based models are able to significantly outperform the benchmark and our proposed method is capable of generating robust models to tackle the dynamic characteristics in the financial application studied. Based upon the promising results obtained, we expect this GA-based method to advance the research in computational intelligence for finance and provide an effective solution to pairs trading for investment in practice.

1. Introduction

In the past decades, due to the inefficacy of traditional statistical approaches, such as regression-based and factor analysis methods for solving difficult financial problems, the methodologies stemming from computational intelligence, including fuzzy theory, artificial neural networks (ANN), support vector machines (SVM), and evolutionary algorithms (EA), have been developed as more effective alternatives to solving the problems in the financial domain [1, 2].

Among the CI-based techniques studied for finance, the models may be classified as two major areas of applications: (1) stock selection, portfolio management, and optimization [3–6] and (2) prediction of financial time series [7, 8]. For the first category, earlier research works include the fuzzy multiple attribute decision analysis for portfolio construction [9]. Zargham and Sayeh [10] employed a fuzzy rule-based system to evaluate a set of stocks for the same task. Chapados and Bengio [11] trained neural networks for estimation and prediction of asset behavior to facilitate decision-making in asset allocation.

In EA applications along this line of research, Becker et al. [12] employed genetic programming (GP) to develop stock ranking models for the U.S. market. Lai et al. [13] used a double-stage GA to select stocks from the Shanghai stock exchange for the time period of years 2001 to 2004. In Lai et al.'s work, ROCE, EPS, PE, and liquidity ratios are used to rank stocks, and they used the GA to compute the optimal percentage of capital assigned to each of the assets. Lai et al. then concluded that their GA-based optimization method is more effective for financial applications than fuzzy or artificial neural networks. Recently, Huang [5] devised a hybrid machine learning-based model to identify promising sets of features and optimal model parameters; Huang's model was demonstrated to be more effective than the benchmark and some traditional statistical methods for stock selection. To improve the performance of the single-objective GA-based models, more recently, Chen et al. [14] proposed a multiobjective GA-based method for the goals of increasing investment return and reducing the risk simultaneously. In that approach, the authors used the nondominated sorting to search for nondominated solutions and showed that the

multiobjective method outperformed the single-objective version proposed by Huang [5].

Another popular study of computational intelligence has been particularly concerning the prediction of financial time series. A certain amount of research employs network learning techniques, including feed-forward, radial basis function or recurrent NN [7], and SVM [8]. Other intelligent methods, such as genetically evolved regression models [15] and inductive fuzzy inference systems [16], were also available in the literature.

Pairs trading [17] is an important research area of computational finance that typically relies on time series data of stock price for investment, in which stocks are bought and sold in pairs for arbitrage opportunities. It is a well-known speculative strategy in the financial markets developed in the 1980s and has been employed as one important long/short equity investment tool by hedge funds and institutional investors [18]. Although there has been a significant amount of CI-based studies in financial applications, reported CI-based research for pairs trading is sparse and lacks serious analysis. To date, many existing works along this line of research rely on traditional statistical methods such as the cointegration approach [19], the Kalman filters [20, 21], and the principle component analysis [18]. In the CI area, Thomaidis et al. [17] employed a method of neural networks for the paired companies of Infosys and Wipro in India and accomplished reasonable return on investment using the pair of stocks. Saks and Maringer [22] used genetic programming for various pairs of stocks in Eurostoxx 50 equities and also found good pair-trading strategies.

Although there exist these previous CI-based studies for pairs trading, they lacked serious analysis such as the method of temporal validation used in [5, 23] for further evaluation of the robustness of the trading systems. In addition, in these previous studies, the trading models were constructed using only two stocks as a trading pair; here, we propose a generalized approach that uses more than two stocks as a trading group for arbitrage in order to further improve the performance of the models. In this study, we also employ the GA for the optimization problems in our proposed arbitrage models. In a past study [23], Huang et al. compared the traditional linear regression and the GA for the task of stock selection and showed that the GA-based model is capable of outperforming the linear regression model. Motivated by this research work, we thus intend to employ the GA to optimize our intelligent system for pairs trading, and the experimental results will show that our proposed GA-based methodology is promising in outperforming the benchmark. Furthermore, in contrast to traditional pairs-trading methods that aim at matching pairs of stocks with similar characteristics, we also show that our method is able to construct working trading models for stocks with different characteristics. In this study, we also investigate the robustness of our proposed method and the results show that our method is indeed effective in generating robust models for the dynamic environment of the pairs-trading problem.

This paper is organized into four sections. Section 2 outlines the method proposed in our study. In Section 3, we describe the research data used in this study and present the experimental results and discussions. Section 4 concludes this paper.

2. Materials and Methods

In this section, we provide the relevant background and descriptions for the design of our pairs-trading systems using the GA for model optimization.

2.1. Pairs Trading. Pairs trading is widely assumed to be the "ancestor" of statistical arbitrage, which is a trading strategy to gain profit from pricing discrepancies in a group of stocks [17]. Traditional decision-making for investment typically relies on fundamentals of companies to assess their value and price their stocks, accordingly. As the true values of the stocks are rarely known, pairs-trading techniques were developed in order to resolve this by investing stock pairs with similar characteristics (e.g., stocks from the same industry). This mutual mispricing between two stocks is theoretically formulated by the notion of spread, which is used to identify the relative positions when an inefficient market results in the mispricing of stocks [18, 21]. As a result, the trading model is usually market-neutral in the sense that it is uncorrelated with the market and may produce a low-volatility investment strategy.

A typical form of pairs trading of stocks operates by selling the stock with a relatively high price and buying the other with a relatively low price at the inception of the trading period, expecting that the higher one will decline while the lower one will rise in the future. The price gap of the two stocks, also known as spread, thus acts as a signal to the open and close positions of the pairs of stocks. During the trading period, position is opened when the spread widens by a certain threshold, and thereafter the positions are closed when spread of the stocks reverts. The objective of this long-short strategy is to profit from the movement of the spread that is expected to revert to its long-term mean.

Consider initial capital X_0, with an interest rate of r per annum and a frequency of compounding n in a year; the capital X after a year may be expressed as

$$X = \left(1 + \frac{r}{n}\right)^n \cdot X_0. \tag{1}$$

If the frequency of compounding n gets arbitrarily large, we have

$$\lim_{n \to \infty} \left(1 + \frac{r}{n}\right)^n = e^r. \tag{2}$$

In the case of continuously compounded return, the process of capital growth is defined as

$$X = e^r \cdot X_0. \tag{3}$$

Therefore, the continuously compounded rate r is calculated by taking the natural logarithm as follows:

$$\ln\left(\frac{X}{X_0}\right) = r, \tag{4}$$

where $\ln(\cdot)$ is the natural log function.

Now consider the two price time series, $P_1(t)$ and $P_2(t)$, of two stocks S_1 and S_2 with similar characteristics, the process of a pairs-trading model can be described as follows [18]:

$$\ln\left(\frac{P_1(t)}{P_1(t_0)}\right) = \alpha(t-t_0) + \beta\ln\left(\frac{P_2(t)}{P_2(t_0)}\right) + \gamma(t), \quad (5)$$

where $\gamma(t)$ is a stationary, mean-reverting process; the drift α is small compared to the fluctuations of $\gamma(t)$ and can be neglected in many applications.

The rationale behind the mean-reverting process is that there exists a long-term equilibrium (mean) for the spread. The investor may bet on the reversion of the current spread to its historical mean by selling and buying an appropriate amount of the pair of the stocks. As (5) shows, one expects the returns of stocks S_1 and S_2 to track each other after controlling for proper β. This model suggests an investment strategy in which one goes long 1 dollar of stock S_1 and short β dollars of stock S_2 if $\gamma(t)$ is small. Conversely, if $\gamma(t)$ is large, one takes an opposite strategy that goes short S_1 and long S_2. As a result, the return of the long-short portfolio may oscillate around a statistical equilibrium.

In real-world practice, the return of the long-short portfolio above for a period of time may be calculated as follows:

$$\mathrm{Ret}_t = \ln\left(\frac{P_1(t)}{P_1(t-1)}\right) - \beta\ln\left(\frac{P_2(t)}{P_2(t-1)}\right), \quad (6)$$

where $P_1(t)$ and $P_1(t-1)$ denote the price of stock S_1 where we are long at time t and $t-1$, respectively; and $P_2(t)$ and $P_2(t-1)$ denote the price of stock S_2 where we are short at time t and $t-1$, respectively.

The pairs-trading method can be generalized to a group of stocks in which mispricing may be identified through a proper combination of assets whose time series is *mean-reverting*. Consider a set of assets, $S_1, \ldots, S_m$, and the corresponding time series of stock prices, $P_1(t), \ldots, P_m(t)$; a statistical mispricing may be considered as a linear combination $B = (\beta_1, \beta_2, \ldots, \beta_m)$ such that

$$\ln\left(\frac{P(t)}{P(t_0)}\right) = \alpha(t-t_0) + \sum_{i=1}^{m}\beta_i\ln\left(\frac{P_i(t)}{P_i(t_0)}\right) + \gamma(t), \quad (7)$$

where $\gamma(t)$ is a mean-reverting process and vector B represents the proportions of one's capital assigned to each asset in the portfolio. Mean reversion in the equation above refers to the assumption that both the high and low prices of the synthetic asset P are temporary and that its price tends to move toward its average price over time.

2.2. Trading Systems

2.2.1. Market Timing Models. In this work, the long-term mean of an asset's price in the mean-reverting process may be modeled by the celebrated moving average [24], which is the average price of an asset in a specified period. Let $P(t)$ be the price of a stock at time t. The moving average at time t, the mean of the prices corresponding to the most recent n time periods, is defined as

$$\mathrm{MA}_n(t) = \frac{1}{n}\sum_{i=1}^{n}P(t-i+1). \quad (8)$$

In this study, we employ the Bollinger Bands [24] to determine if the spread of a pair of stocks departs from its dynamic average value. Typically, the Bollinger Bands prescribe two volatility bands placed above and below a moving average, in which volatility may be defined as a multiple of the standard deviation of the prices in the past. Formally the Bollinger Bands can be defined as follows:

$$\begin{aligned}\mathrm{MB}_n(t) &= \mathrm{MA}_n(t);\\ \mathrm{UB}_n(t) &= \mathrm{MB}_n(t) + k*\sigma_n(t);\\ \mathrm{LB}_n(t) &= \mathrm{MB}_n(t) - k*\sigma_n(t),\end{aligned} \quad (9)$$

where $\sigma_n(t)$ is the standard deviation of the prices, at time t, over the past n time periods; $k \in R$ is a parameter used to control the width of the upper and lower bands to the moving average.

An important component of a successful trading system is to construct models for market timing that prescribe meaningful entry and exit points in the market. In this study, we will use the moving averages and Bollinger Bands to develop a trading system, which is described in the next subsection.

2.2.2. Trading Strategy and Performance Evaluation. We calculate the spread for the synthetic asset generated by m stocks as

$$P(t) = \sum_{i=1}^{m}\beta_i P_i(t), \quad (10)$$

where $P_i(t)$, $i = 1, \ldots, m$, is the price of stock i at time t, and β_i's are the model parameters of generalized pairs trading to be estimated.

In this work, we designate the trading strategy for one to buy (sell) the spread right after it gets x standard deviations below (above) its mean value and the position is closed right after the spread gets closer than y standard deviations to its mean, where $x, y \in R$ and $x > y > 0$.

Here we evaluate the performance of a trading system in terms of its compounded return, which is to be determined by the relevant parameters of the trading models employed. We first define the return of a trading system for the lth trade as $R_l(\theta) \in R$, where θ denotes the set of the model parameters. Then the performance metric we use here is through the total cumulative (compounded) return, R_c, where R_c is defined by the product of the returns over z consecutive trades as

$$R_c = \prod_{l=1}^{z}R_l. \quad (11)$$

Therefore, in the process of capital growth, the capital X_z at the end of z trades is

$$X_z = R_c X_0, \quad (12)$$

where X_0 represents the initial capital.

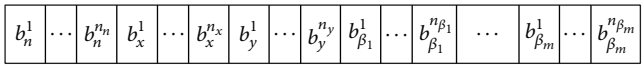


Figure 1: Chromosome encoding.

2.3. Optimization of Trading Systems. Given the market timing and pairs-trading models, the performance of a trading system shall be enhanced by suitable values of the corresponding model parameters. For the market timing models, the parameters include the period n for the moving average and parameters x and y for the Bollinger Bands that controls the multiples of the standard deviations of the moving average for entry and exit points. For the pairs-trading model, the parameters consist of the set of the weighting terms (β_i's) in the syntactic asset from (10). In this study, we propose using genetic algorithms (GA) for the search of optimal parameters of the trading system. We will describe the basics of GA as well as our proposed optimization scheme in the following.

Genetic algorithms [25] have been used as computational simulation models of natural evolutionary systems and as adaptive algorithms for solving complex optimization problems in the real world. The core of this class of algorithms lies in the production of new genetic structures, along the course of evolution, that provide innovations to solutions for the problem. Typically, the GA operate on an evolving population of artificial agents whose composition can be as simple as a binary string that encodes a solution to the problem at hand and a phenotype that represents the solution itself. In each iteration, a new generation is created by applying crossover and mutation to candidates selected as the parents. Evolution occurs by iterated stochastic variation of genotypes and selection of the fit phenotypes in an environment based on how well the individual solutions solve a problem.

In our proposed encoding design, the composition of a chromosome is devised to consist of four portions that encode the period parameter n for the moving average, the multiples x and y of standard deviations for the Bollinger Bands, and the set of the weighting coefficients (β_i's) for the pairs-trading model from (10). Here we use the binary coding scheme to represent a chromosome in the GA. In Figure 1, loci b_n^1 through $b_n^{n_n}$ represent the encoding for the period n of moving average. Loci b_x^1 through $b_x^{n_x}$ and b_y^1 through $b_y^{n_y}$ represent the encoding of x and y for the Bollinger Bands, respectively. Finally, loci $b_{\beta_i}^1$ through $b_{\beta_i}^{n_{\beta_i}}$ represent the encoding of the weighting coefficient β_i, $i = 1, \ldots, m$.

In our encoding scheme, the chromosome representing the genotypes of parameters is to be transformed into the phenotype by (13) below for further fitness computation. The precision representing each parameter depends on the number of bits used to encode it in the chromosome, which is determined as follows:

$$y = \min_y + \frac{d}{2^i - 1} \times \left(\max_y - \min_y\right), \tag{13}$$

where y is the corresponding phenotype for the particular parameter; $\min_y$ and $\max_y$ are the minimum and maximum values of the parameter; d is the corresponding decimal value (d being truncated to integers if the parameter is of integer type), and l is the length of the block used to encode the parameter in the chromosome.

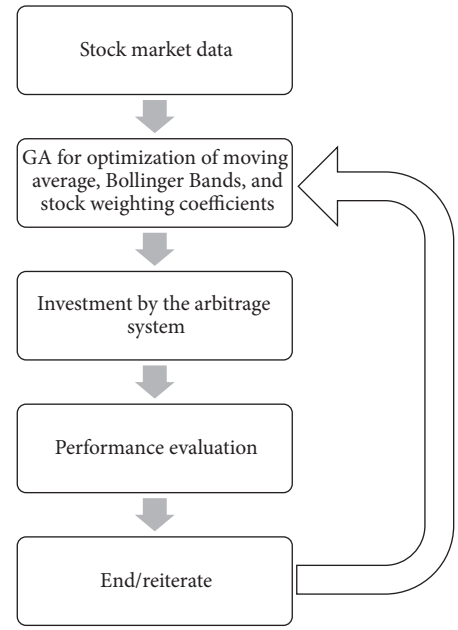


Figure 2: Flow chart of the GA-based arbitrage system.

With this scheme, we define the fitness function of a chromosome as the annualized return of the trading system over h years of investment:

$$\text{fitness} = \sqrt[h]{R_c}, \tag{14}$$

where R_c is the total cumulative return computed by (11).

Our overall GA-based arbitrage system is a multistage process, including the simultaneous optimization on the weighting coefficients for stocks, the period for the moving average, and the width of the Bollinger Bands. The input to the system is the time series datasets of stock price. For any given combinations of model parameters of the moving average, Bollinger Bands, and the weighting coefficients of stocks, we employ the pairs-trading arbitrage system for investment. In this work, the timing for trading is designated as buying (selling) the spread right after it gets to a certain distance (measured by standard deviations to the average) below (above) the average and the position is then closed right after the spread gets closer to the mean. The stocks to be long or short are determined by the weighting terms (β_i's) in the syntactic asset from (10). We then compute the corresponding returns for the performance evaluation of the system. In this study, the GA is used as the optimization tool for simultaneous optimization of these model parameters. The final output is a set of models parameters (optimized by the GA) that prescribes the pairs-trading and timing models. The flowchart of this GA-based trading system is summarized in Figure 2.

Table 1: The 10 semiconductor stocks used in this study.

Ticker	Name (Chinese)	Name (English)
2325	矽品	Siliconware Precision Industries Co, Ltd.
2330	臺積電	Taiwan Semiconductor Manufacturing
2303	聯電	United Microelectronics Corp.
2311	日月光	Advanced Semiconductor Engineering, Inc.
2337	旺宏	Macronix International Company, Ltd.
2301	光寶科	Lite-On Technology Corp.
2308	台達電	Delta Electronics, Inc.
2409	友達	AU Optronics Corporation
2451	創見	Transcend Information, Inc.
2454	聯發科	MediaTek Inc.

3. Results and Discussion

In this section we examine the performance of our proposed method for pair-trading systems. We use two sets of stocks listed in the Taiwan Stock Exchange for illustration: (1) the set of 10 stocks with similar characteristics from the semiconductor industry, which is the most important industrial sector in Taiwan over the past two decades, and (2) the set of the 10 stocks with largest market capitalization from various sectors, which denote distinct industrial characteristics in Taiwan.

3.1. 10 Stocks from the Semiconductor Industry. The daily returns of the 10 semiconductor stocks in Taiwan from years 2003 to 2012 were used to examine the performance of the GA-optimized trading system. Table 1 shows the 10 stocks used for this subsection. Figure 3 displays an illustration of the best-so-far curve for the accumulated return (i.e., the total cumulative return) attained by the GA over 50 generations. (In order to study the quality of solutions over time, a traditional performance metric for the GA is the "best-so-far" curve that plots the fitness of the best individual that has been seen thus far by generation n, i.e., a point in the search space that optimizes the objective function thus far. In addition, in this study, the GA experiments employ a binary tournament selection [26], one-point crossover, and mutation rates of 0.7 and 0.005, resp. We also use 10 bits to encode each variable in the chromosome and use 50 individuals for the size of the population in each generation.) This figure shows how the GA searches for the solutions over the course of evolution to gradually improve the performance of the trading system.

Figure 4 displays an illustration of the accumulated return of the benchmark and that of our GA-based model. (In this study, the benchmark is defined as the traditional buy-and-hold method where we allocate one's capital in equal proportion to each stock and the accumulated return is calculated as the product of the average daily returns of all the 10 stocks over the 10 years; i.e., an investor invests all the capital in the stocks initially and sell all of them only at the end of the course of investment.) This figure shows that the GA-based model gradually outperforms the benchmark and the performance discrepancy becomes quite significant at the end of year 2012. As opposed to the buy-and-hold method that allocates one's capital in equal proportions to each stock, the GA proactively searches for the optimal proportions for long or short positions for each asset in order to construct the spread by (10). In addition, the GA also searches for the optimal timing for buying and shorting the stocks dynamically using the Bollinger Bands. In our study here, the weighting coefficients for the proportions of capital allocated to stocks, the period for the moving average, and the width of the Bollinger Bands are optimized simultaneously.

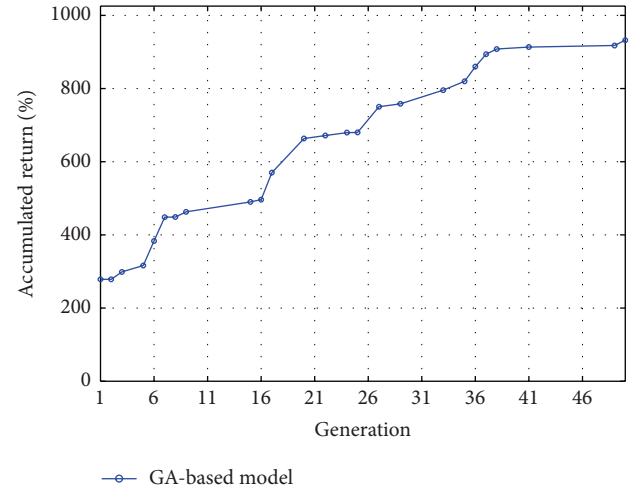


Figure 3: An illustration for the best-so-far curve by the GA.

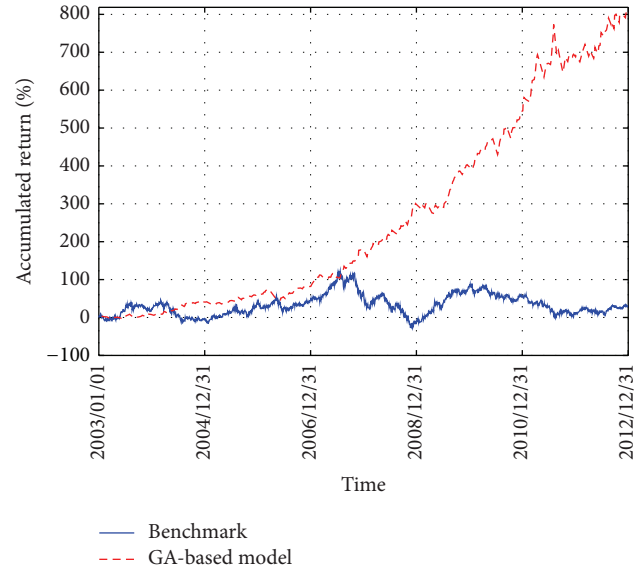


Figure 4: Accumulated return of the benchmark versus the GA-based model for the 10 semiconductor stocks from years 2003 to 2012.

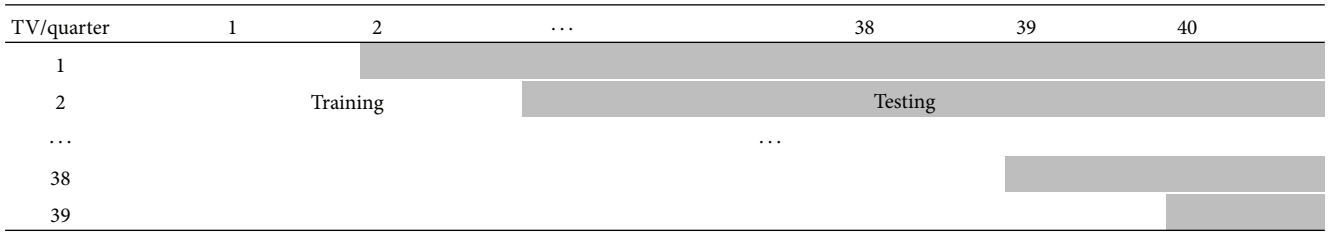


Figure 5: Temporal validation.

As a result, in our proposed methodology, a trading system optimized by the GA is a composite of optimal arbitrage and market timing models. Thus, one may expect the GA to be advantageous to the construction of the arbitrage systems and Figure 4 indeed shows that the GA-based model outperforms the benchmark in the long run. Therefore, these results shed some light on how the optimization by the GA may be advantageous to the pairs-trading model.

In order to further examine the validity of our proposed method, statistical validation on the models is conducted in this study. In reality, the learned model using the training data has to be tested by unseen data. Here, as shown in Figure 5, we use the stock data of the first several quarters to train the model, and the remaining data is used for testing. This setup is to provide a set of temporal validations to examine the effectiveness of the models in the dynamic environment of financial problems, which is different from the regular cross-validation procedure where the process of data being split into two independent sets is randomly repeated several times without taking into account the data's temporal order. However, in the financial study here, temporal order is critical since one would like to use all available data so far to train the model and to apply the models in the future for profits.

In the training phase of each TV, we conduct 50 runs for the GA and the best model learned from each run is examined in the testing phase. In both of the training and testing phases, the cumulative total return (accumulated return) of a model over the quarters is calculated and the corresponding annualized return is computed by (14). The annualized returns of the best 50 models in each TV are then averaged and displayed for the training and testing phases in Table 2. In this table, we also provide the annualized benchmark return for further comparison with the GA-based models, where the cumulative total return for the benchmark is calculated from the product of the average quarterly returns of the 10 semiconductor stocks over the period of time in training or testing, and the corresponding annualized return is again computed by (14).

In Table 2, an inspection on the means of annualized model returns shows that in all the 39 TVs of the training case the GA-based method outperforms the benchmark. For the testing phase, in 30 out of 39 cases the GA-based method outperforms the benchmark. Figure 6 further displays a visual gist on this performance discrepancy of the two methods in the testing phase. As can be seen, in most of the TVs, the annualized return of the GA-based model is larger than that of the benchmark. These results thus demonstrate our GA-based method is promising for solving the pairs-trading problem.

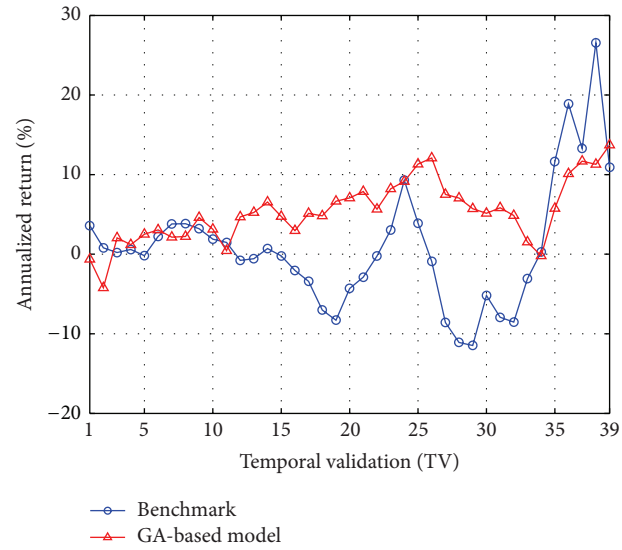


Figure 6: Averaged annualized return of the top 50 GA-based models versus the benchmark (in each TV of the testing phase) for the 10 semiconductor stocks from years 2003 to 2012.

3.2. 10 Stocks with the Largest Market Capitalization. Next we use the 10 stocks of the largest market capitalization listed in the Taiwan Stock Exchange to further examine our proposed method. The daily returns of stocks from years 2003 to 2012 were again used for the optimization task by the GA. Table 3 shows the 10 stocks with the largest market cap used in this study.

Figure 7 displays an illustration of the accumulated return of the benchmark (which is again defined as the product of the average daily returns of the 10 largest market cap stocks over the 10 years) and that of our GA-based model. As can be seen, the GA-based model gradually outperforms the benchmark over the course of investment during years 2003 to 2012, and the performance discrepancy becomes significant at the end of year 2012. This figure thus illustrates how the GA-based model may outperform the benchmark in the long run.

For the temporal validation, by the same procedure used in the previous subsection, Table 4 shows the annualized benchmark return and the average of the annualized model

Table 2: Comparisons of the annualized returns of the GA-based model and the benchmark for the 10 semiconductor stocks from years 2003 to 2012.

TV	Training period	Annualized benchmark return	Mean of annualized model return	Standard deviation	Testing period	Annualized benchmark return	Mean of annualized model return	Standard deviation
1	2003Q1	−27.23%	157.55%	0.0494	2003Q2–2012Q4	3.60%	0.01%	0.5125
2	2003Q1–2003Q2	36.06%	137.35%	0.1117	2003Q3–2012Q4	0.80%	−4.42%	0.5494
3	2003Q1–2003Q3	39.79%	119.60%	0.1119	2003Q4–2012Q4	0.21%	3.49%	0.5392
4	2003Q1–2003Q4	20.67%	85.98%	0.1498	2004Q1–2012Q4	0.59%	2.05%	0.4707
5	2003Q1–2004Q1	24.25%	72.06%	0.1553	2004Q2–2012Q4	−0.19%	2.55%	0.4833
6	2003Q1–2004Q2	5.26%	59.23%	0.1626	2004Q3–2012Q4	2.22%	5.14%	0.7364
7	2003Q1–2004Q3	−2.64%	55.70%	0.3206	2004Q4–2012Q4	3.80%	4.49%	0.6824
8	2003Q1–2004Q4	−1.33%	53.64%	0.3889	2005Q1–2012Q4	3.83%	5.17%	0.8286
9	2003Q1–2005Q1	0.03%	45.19%	0.3904	2005Q2–2012Q4	3.20%	2.58%	0.6446
10	2003Q1–2005Q2	4.49%	42.85%	0.4555	2005Q3–2012Q4	1.88%	3.24%	0.6986
11	2003Q1–2005Q3	5.36%	42.02%	0.4171	2005Q4–2012Q4	1.48%	1.58%	0.4788
12	2003Q1–2005Q4	11.17%	34.85%	0.3509	2006Q1–2012Q4	−0.77%	4.70%	0.5345
13	2003Q1–2006Q1	9.63%	34.61%	0.2917	2006Q2–2012Q4	−0.57%	5.29%	0.5652
14	2003Q1–2006Q2	6.65%	32.14%	0.2780	2006Q3–2012Q4	0.71%	5.92%	0.5332
15	2003Q1–2006Q3	7.54%	31.70%	0.3518	2006Q4–2012Q4	−0.22%	4.58%	0.4451
16	2003Q1–2006Q4	9.38%	30.56%	0.3527	2007Q1–2012Q4	−2.03%	4.16%	0.3340
17	2003Q1–2007Q1	11.74%	30.21%	0.4362	2007Q2–2012Q4	−3.41%	5.25%	0.3628
18	2003Q1–2007Q2	15.86%	27.98%	0.5660	2007Q3–2012Q4	−7.02%	8.11%	0.6571
19	2003Q1–2007Q3	16.65%	31.03%	0.7558	2007Q4–2012Q4	−8.30%	7.48%	0.4423
20	2003Q1–2007Q4	10.80%	32.33%	0.7995	2008Q1–2012Q4	−4.31%	7.86%	0.5680
21	2003Q1–2008Q1	7.98%	31.33%	1.0381	2008Q2–2012Q4	−2.88%	8.24%	0.3712
22	2003Q1–2008Q2	5.50%	35.78%	1.4992	2008Q3–2012Q4	−0.23%	7.87%	0.3083
23	2003Q1–2008Q3	2.35%	34.33%	1.9692	2008Q4–2012Q4	3.05%	6.84%	0.2409
24	2003Q1–2008Q4	−2.29%	34.66%	1.9921	2009Q1–2012Q4	9.29%	8.63%	0.2430
25	2003Q1–2009Q1	1.88%	31.07%	1.7270	2009Q2–2012Q4	3.90%	9.78%	0.2099
26	2003Q1–2009Q2	4.43%	29.88%	1.6957	2009Q3–2012Q4	−0.90%	10.43%	0.2683
27	2003Q1–2009Q3	8.62%	31.06%	2.1604	2009Q4–2012Q4	−8.58%	8.68%	0.2157
28	2003Q1–2009Q4	9.09%	31.06%	2.3144	2010Q1–2012Q4	−11.09%	6.66%	0.1684
29	2003Q1–2010Q1	8.39%	31.53%	2.3984	2010Q2–2012Q4	−11.47%	5.74%	0.2323
30	2003Q1–2010Q2	5.83%	30.06%	2.5543	2010Q3–2012Q4	−5.19%	5.89%	0.1809
31	2003Q1–2010Q3	5.83%	27.71%	2.2193	2010Q4–2012Q4	−7.93%	6.05%	0.1847
32	2003Q1–2010Q4	5.80%	27.26%	2.7974	2011Q1–2012Q4	−8.53%	5.48%	0.1443
33	2003Q1–2011Q1	3.86%	29.38%	2.8889	2011Q2–2012Q4	−3.09%	1.32%	0.1056
34	2003Q1–2011Q2	3.11%	28.69%	3.3213	2011Q3–2012Q4	0.29%	0.20%	0.1043
35	2003Q1–2011Q3	1.82%	27.14%	2.7034	2011Q4–2012Q4	11.66%	5.24%	0.0842
36	2003Q1–2011Q4	1.22%	27.73%	3.7890	2003Q1–2012Q4	18.89%	11.33%	0.0745
37	2003Q1–2012Q1	2.03%	25.98%	3.3508	2003Q1–2012Q4	13.27%	12.24%	0.0604
38	2003Q1–2012Q2	1.58%	25.08%	3.2671	2003Q1–2012Q4	26.57%	11.28%	0.0582
39	2003Q1–2012Q3	2.50%	25.38%	3.6955	2012Q4	10.91%	12.68%	0.0414

Table 3: The 10 largest market cap stocks used in this study.

Ticker	Name (Chinese)	Name (English)
2330	臺積電	Taiwan Semiconductor Manufacturing
2317	鴻海	Hon Hai Precision
2454	聯發科	MediaTek Inc.
1301	臺塑	Formosa Plastics
1303	南亞	Nan Ya Plastics
1326	台化	Formosa Chemicals
2412	中華電	Chunghwa Telecom
2882	國泰金	Cathay Financial Holding
2308	台達電	Delta Electronics, Inc.
2008	高興昌	Kao Hsing Chang Iron

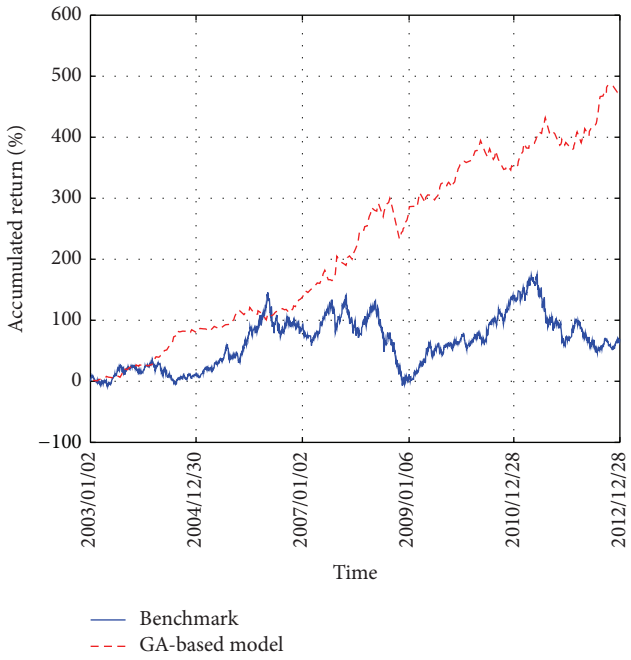


Figure 7: Accumulated return of the benchmark versus the GA-based model for the 10 largest market cap stocks from years 2003 to 2012.

returns for the training and testing cases. As can be seen from the means of the annualized model returns in the training case, the GA-based method outperforms the benchmark in all the 39 TVs. For the testing phase, in 29 out of 39 cases the GA-based method outperforms the benchmark, as well. Figure 8 then displays the results in Table 4 for each TV in the testing phase. An inspection of Figure 8 thus shows that, in 29 out of 39 TVs, the GA-based models outperform the benchmark in terms of annualized returns.

3.3. Model Robustness. Finally, we examine the robustness of the models generated by our method using the measure of

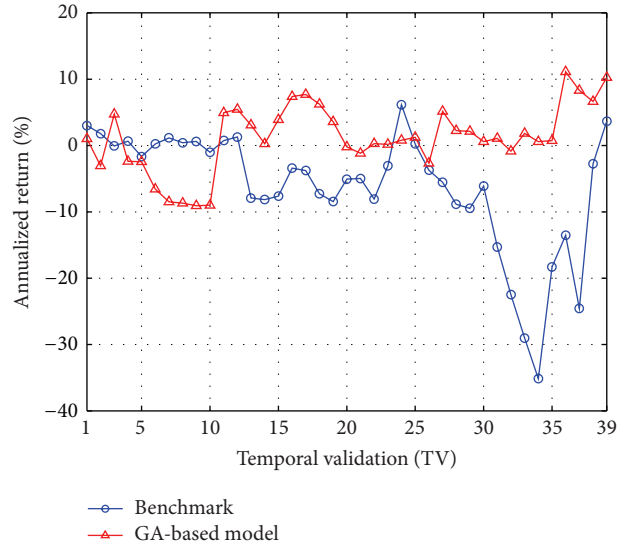


Figure 8: Averaged annualized return of the top 50 GA-based models versus the benchmark (in each TV of the testing phase) for the 10 largest market cap stocks from years 2003 to 2012.

precision studied in [5], which is defined as

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{15}$$

In this definition, TP and FP denote the number of true positives and false positives, respectively. In this study, a true positive occurs when a model outperforms the benchmark in training, and it later turns out to outperform the benchmark in testing, as well; otherwise, the model generates a false positive. This statistic is an important metric that indicates whether our proposed method can generate robust models when the problem is in a dynamic environment, such as the financial problem studied here.

Typically, if a method generates a model that outperforms the benchmark in the training phase, one would like the model to continue to outperform the benchmark in the testing phase. Therefore, if our proposed method is able to generate many true positives that leads to high precision, it is an indication that our method is effective in generating robust models. Table 5 displays the results of precision for the 10 semiconductor and largest market cap stocks. As can be seen, the results show that the precision of our proposed method is more than 0.7 in both cases, thereby indicating that our proposed method is indeed effective.

4. Conclusions

In this paper, we presented a GA-based methodology for the application of pairs trading in computational finance. In order to examine the validity of the proposed methodology, we conducted a statistical validation on the learned models to account for the temporal order and dynamic characteristics of the stock data, which is critical for the real-world investment as practically one expects the models constructed to gain profits in the future. Through the optimization of parameters

Table 4: Comparisons of the annualized returns of the GA-based model and the benchmark for the 10 largest market cap stocks from years 2003 to 2012.

TV	Training period	Annualized benchmark return	Mean of annualized model return	Standard deviation	Testing period	Annualized benchmark return	Mean of annualized model return	Standard deviation
1	2003Q1	−9.13%	133.86%	0.3965	2003Q2–2012Q4	33.58%	1.03%	0.0472
2	2003Q1–2003Q2	18.81%	109.67%	0.1066	2003Q3–2012Q4	18.51%	−3.06%	0.0348
3	2003Q1–2003Q3	33.87%	86.87%	0.1665	2003Q4–2012Q4	−0.69%	4.74%	0.0571
4	2003Q1–2003Q4	26.40%	54.91%	0.0853	2004Q1–2012Q4	6.10%	−2.38%	0.0344
5	2003Q1–2004Q1	31.57%	52.05%	0.0939	2004Q2–2012Q4	−13.55%	−2.43%	0.0558
6	2003Q1–2004Q2	16.71%	60.24%	0.1204	2004Q3–2012Q4	1.99%	−6.57%	0.0407
7	2003Q1–2004Q3	11.80%	62.13%	0.0971	2004Q4–2012Q4	9.94%	−8.50%	0.0273
8	2003Q1–2004Q4	15.29%	55.99%	0.1029	2005Q1–2012Q4	3.50%	−8.68%	0.0305
9	2003Q1–2005Q1	12.64%	49.95%	0.0850	2005Q2–2012Q4	4.91%	−9.07%	0.0449
10	2003Q1–2005Q2	15.71%	40.65%	0.0636	2005Q3–2012Q4	−7.39%	−9.01%	0.0498
11	2003Q1–2005Q3	6.79%	37.80%	0.0596	2005Q4–2012Q4	5.58%	4.94%	0.0401
12	2003Q1–2005Q4	12.54%	35.57%	0.0478	2006Q1–2012Q4	9.46%	5.46%	0.0724
13	2003Q1–2006Q1	29.21%	30.42%	0.0591	2006Q2–2012Q4	−42.67%	3.09%	0.0614
14	2003Q1–2006Q2	32.17%	39.09%	0.0536	2006Q3–2012Q4	−42.45%	0.26%	0.0623
15	2003Q1–2006Q3	23.30%	32.03%	0.0724	2006Q4–2012Q4	−39.08%	3.91%	0.0655
16	2003Q1–2006Q4	9.71%	28.78%	0.0284	2007Q1–2012Q4	−18.77%	7.38%	0.0341
17	2003Q1–2007Q1	10.42%	28.97%	0.0271	2007Q2–2012Q4	−19.82%	7.71%	0.0382
18	2003Q1–2007Q2	15.69%	27.25%	0.0357	2007Q3–2012Q4	−33.97%	6.24%	0.0324
19	2003Q1–2007Q3	16.14%	29.75%	0.0323	2007Q4–2012Q4	−37.09%	3.59%	0.0397
20	2003Q1–2007Q4	11.22%	31.29%	0.0383	2008Q1–2012Q4	−22.89%	−0.22%	0.0292
21	2003Q1–2008Q1	9.96%	33.59%	0.0436	2008Q2–2012Q4	−21.45%	−1.16%	0.0466
22	2003Q1–2008Q2	13.56%	28.81%	0.0756	2008Q3–2012Q4	−31.51%	0.29%	0.0438
23	2003Q1–2008Q3	8.29%	27.84%	0.0506	2008Q4–2012Q4	−12.44%	0.18%	0.0357
24	2003Q1–2008Q4	1.30%	24.99%	0.0535	2009Q1–2012Q4	26.95%	0.81%	0.0501
25	2003Q1–2009Q1	5.04%	23.92%	0.0516	2009Q2–2012Q4	0.86%	1.23%	0.0725
26	2003Q1–2009Q2	7.37%	25.11%	0.0395	2009Q3–2012Q4	−12.45%	−2.67%	0.0802
27	2003Q1–2009Q3	6.73%	24.79%	0.0252	2009Q4–2012Q4	−16.92%	5.16%	0.0357
28	2003Q1–2009Q4	7.63%	25.15%	0.0215	2010Q1–2012Q4	−24.31%	2.24%	0.0326
29	2003Q1–2010Q1	7.14%	25.32%	0.0201	2010Q2–2012Q4	−23.97%	2.14%	0.0378
30	2003Q1–2010Q2	7.28%	23.47%	0.0337	2010Q3–2012Q4	−14.58%	0.59%	0.0753
31	2003Q1–2010Q3	9.81%	21.78%	0.0373	2010Q4–2012Q4	−31.17%	1.08%	0.0947
32	2003Q1–2010Q4	11.29%	20.65%	0.0395	2011Q1–2012Q4	−39.94%	−0.85%	0.0970
33	2003Q1–2011Q1	12.29%	21.09%	0.0343	2011Q2–2012Q4	−45.11%	1.84%	0.0532
34	2003Q1–2011Q2	13.44%	20.05%	0.0388	2011Q3–2012Q4	−47.78%	0.58%	0.0600
35	2003Q1–2011Q3	7.56%	21.41%	0.0359	2011Q4–2012Q4	−22.33%	0.75%	0.1054
36	2003Q1–2011Q4	8.08%	18.04%	0.0389	2003Q1–2012Q4	−13.53%	11.13%	0.0909
37	2003Q1–2012Q1	6.51%	19.60%	0.0388	2003Q1–2012Q4	−19.06%	8.32%	0.0801
38	2003Q1–2012Q2	4.77%	19.41%	0.0253	2003Q1–2012Q4	−1.38%	6.64%	0.1144
39	2003Q1–2012Q3	3.43%	19.58%	0.0171	2012Q4	0.91%	10.26%	0.1145

Table 5: Precision for the 10 semiconductor and largest market cap stocks.

	10 semiconductor stocks	10 largest market cap stocks
Precision	0.7692	0.7180

of the trading models for a group of stocks, the experimental results showed that our GA-based method is able to significantly outperform the benchmark and can generate robust models for pairs trading. We thus expect this GA-based method to advance the research in computational intelligence for financial applications and provide a promising solution to pairs trading.

Conflict of Interests

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

Acknowledgments

This work is fully supported by the National Science Council, Taiwan, under Grant no. MOST 103-2221-E-390-019. The authors would also like to thank Professor Chih-Hsiang Chang for his generosity in providing the financial data.

References

[1] A. M. Farley and S. Jones, "Using a genetic algorithm to determine an index of leading economic indicators," *Computational Economics*, vol. 7, no. 3, pp. 163–173, 1994.

[2] A. Mochón, D. Quintana, Y. Sáez, and P. Isasi, "Soft computing techniques applied to finance," *Applied Intelligence*, vol. 29, no. 2, pp. 111–115, 2008.

[3] K.-J. Kim and I. Han, "Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index," *Expert Systems with Applications*, vol. 19, no. 2, pp. 125–132, 2000.

[4] Y. Becker, P. Fei, and A. Lester, "Stock selection—an innovative application of genetic programming methodology," in *Genetic Programming Theory and Practice IV*, R. Riolo, T. Soule, and B. Worzel, Eds., vol. 5 of *Genetic and Evolutionary Computation*, chapter 12, pp. 315–334, Springer, Ann Arbor, Mich, USA, 2006.

[5] C.-F. Huang, "A hybrid stock selection model using genetic algorithms and support vector regression," *Applied Soft Computing Journal*, vol. 12, no. 2, pp. 807–818, 2012.

[6] X. Z. Zhang, Y. Hu, K. Xie, S. Y. Wang, E. W. T. Ngai, and M. Liu, "A causal feature selection algorithm for stock prediction modeling," *Neurocomputing*, vol. 14, pp. 48–59, 2014.

[7] G. Zhang, B. E. Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks: the state of the art," *International Journal of Forecasting*, vol. 14, no. 1, pp. 35–62, 1998.

[8] C.-L. Huang and C.-Y. Tsai, "A hybrid SOFM-SVR with a filter-based feature selection for stock market forecasting," *Expert Systems with Applications*, vol. 36, no. 2, pp. 1529–1539, 2009.

[9] T.-C. Chu, C.-T. Tsao, and Y.-R. Shiue, "Application of fuzzy multiple attribute decision making on company analysis for stock selection," in *Proceedings of the 1996 Asian Fuzzy Systems Symposium on Soft Computing on Intelligent Systems and Information Processing*, pp. 509–514, Kenting, Taiwan, December 1996.

[10] M. R. Zargham and M. R. Sayeh, "A Web-based information system for stock selection and evaluation," in *Proceedings of the 1st International Workshop on Advance Issues of E-Commerce and Web-Based Information Systems*, pp. 81–83, 1999.

[11] N. Chapados and Y. Bengio, "Cost functions and model combination for VaR-based asset allocation using neural networks," *IEEE Transactions on Neural Networks*, vol. 12, no. 4, pp. 890–906, 2001.

[12] Y. Becker, P. Fei, and A. Lester, "Stock selection: an innovative application of genetic programming methodology," in *Genetic Programming Theory and Practice IV*, R. Riolo, T. Soule, and B. Worzel, Eds., Springer, New York, NY, USA, 2007.

[13] K. K. Lai, L. Yu, S. Wang, and C. Zhou, "A double-stage genetic optimization algorithm for portfolio selection," in *Neural Information Processing: Proceedings of the 13th International Conference, ICONIP 2006, Hong Kong, China, October 3–6, 2006, Part III*, vol. 4234 of *Lecture Notes in Computer Science*, pp. 928–937, Springer, Berlin, Germany, 2006.

[14] S.-S. Chen, C.-F. Huang, and T.-P. Hong, "A multi-objective genetic model for stock selection," in *Proceedings of the 27th Annual Conference of the Japanese Society for Artificial Intelligence*, 2013.

[15] G. G. Szpiro, "Forecasting chaotic time series with genetic algorithms," *Physical Review E*, vol. 55, no. 3, pp. 2557–2568, 1997.

[16] A. Fiordaliso, "A nonlinear forecasts combination method based on Takagi-Sugeno fuzzy systems," *International Journal of Forecasting*, vol. 14, no. 3, pp. 367–379, 1998.

[17] N. S. Thomaidis, N. Kondakis, and G. D. Dounias, "An intelligent statistical arbitrage trading system," in *Advances in Artificial Intelligence*, vol. 3955 of *Lecture Notes in Computer Science*, pp. 596–599, Springer, Berlin, Germany, 2006.

[18] M. Avellaneda and J.-H. Lee, "Statistical arbitrage in the US equities market," *Quantitative Finance*, vol. 10, no. 7, pp. 761–782, 2010.

[19] E. Gatev, W. N. Goetzmann, and K. G. Rouwenhorst, "Pairs trading: performance of a relative-value arbitrage rule," *Review of Financial Studies*, vol. 19, no. 3, pp. 797–827, 2006.

[20] C. L. Dunis, G. Giorgioni, J. Laws, and J. Rudy, "Statistical arbitrage and high-frequency data with an application to Eurostoxx 50 equities," CIBEF Working Papers, CIBEF, 2010.

[21] R. J. Elliott, J. van der Hoek, and W. P. Malcolm, "Pairs trading," *Quantitative Finance*, vol. 5, no. 3, pp. 271–276, 2005.

[22] P. Saks and D. Maringer, "Genetic programming in statistical arbitrage," in *Applications of Evolutionary Computing*, vol. 4974 of *Lecture Notes in Computer Science*, pp. 73–82, Springer, Berlin, Germany, 2008.

[23] C.-F. Huang, T.-N. Hsieh, B. R. Chang, and C.-H. Chang, "A comparative study of stock scoring using regression and genetic-based linear models," in *Proceedings of the IEEE International Conference on Granular Computing (GrC '11)*, pp. 268–273, IEEE, Kaohsiung, Taiwan, November 2011.

[24] J. Murphy, *Technical Analysis of Financial Markets*, New York Institute of Finance, New York, NY, USA, 1999.

[25] J. H. Holland, *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor, Mich, USA, 1975.

[26] D. E. Goldberg and K. Deb, "A comparative analysis of selection schemes used in genetic algorithms," in *Foundation of Genetic Algorithms*, pp. 69–93, 1991.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 796276, 21 pages
http://dx.doi.org/10.1155/2015/796276

Research Article

Symmetry Based Automatic Evolution of Clusters: A New Approach to Data Clustering

Singh Vijendra[1] and Sahoo Laxman[2]

[1] *Department of Computer Science and Engineering, Faculty of Engineering and Technology, Mody University of Science and Technology, Lakshmangarh, Rajasthan 332311, India*
[2] *School of Computer Engineering, KIIT University, Bhubaneswar 751024, India*

Correspondence should be addressed to Singh Vijendra; vsingh.fet@gmail.com

Received 1 November 2014; Accepted 10 January 2015

Academic Editor: Rahib H. Abiyev

We present a multiobjective genetic clustering approach, in which data points are assigned to clusters based on new line symmetry distance. The proposed algorithm is called multiobjective line symmetry based genetic clustering (MOLGC). Two objective functions, first the Davies-Bouldin (DB) index and second the line symmetry distance based objective functions, are used. The proposed algorithm evolves near-optimal clustering solutions using multiple clustering criteria, without a priori knowledge of the actual number of clusters. The multiple randomized K dimensional (Kd) trees based nearest neighbor search is used to reduce the complexity of finding the closest symmetric points. Experimental results based on several artificial and real data sets show that proposed clustering algorithm can obtain optimal clustering solutions in terms of different cluster quality measures in comparison to existing SBKM and MOCK clustering algorithms.

1. Introduction

Clustering is one of the most common unsupervised data mining methods to explore the hidden structures embedded in a data set [1]. Clustering gives rise to a variety of information granules whose use reveals the structure of data [2]. Clustering has been effectively applied in a variety of engineering and scientific disciplines [3]. In order to mathematically identify clusters in a data set, it is usually necessary to first define a measure of similarity or proximity which will establish a rule for assigning patterns to the domain of a particular cluster center. Symmetry is considered as a preattentive feature that enhances recognition and reconstruction of shapes and objects. However, the exact mathematical definition of symmetry, such as Miller [4], is inadequate to describe and quantify symmetry found in the natural world or those found in the visual world. Since symmetry is so common in the abstract and in the nature, it is reasonable to assume that some kind of symmetry occur in the structures of clusters. The immediate problem is how to find a way to measure symmetry. Zabrodsky et al. [5] have proposed a kind of symmetry distance to detect symmetry in a figure extracted from an image. Their basic strategy is to choose the symmetry that is the closest to the figure measured by an appropriate measure, in which they adopt the minimum sum of the squared distances over which the vertices must be removed to impose the assumed symmetry. It follows that we need an algorithm for effectively imposing a given symmetry with a minimum displacement [6]. A new type of nonmetric distance, based on point symmetry, is proposed which is used in a K-means based clustering algorithm, referred to as symmetry based K-means (SBKM) algorithm [7]. SBKM will fail for some data sets where the clusters themselves are symmetrical with respect to some intermediate point. This work is extended in Chung and Lin [8] to overcome some of limitations existing in SBKM. These symmetry based clustering techniques adopted the concept of K-means for discovering clusters. The K-means [9] algorithm is one of the more widely used clustering algorithms. However, it is well known that K-means algorithm is sensitive to

the initial cluster centers and easy to get stuck at the local optimal solutions. Second important problem in partitioning clustering is to find a partition of the given data, with a specified number of clusters that minimizes the total within cluster variation. Unfortunately in many real life cases the number of clusters in a data set is not known a priori.

In order to overcome the limitation of being easy to get stuck at the local optimal solutions, some attempts have been made to use genetic algorithms for clustering data sets [10–12]. To overcome the problem of automatic cluster determination from the data sets. Recently, many automatic clustering techniques have been introduced. These automatic clustering techniques are based on genetic algorithm methods and Differential Evolution (DE) methods. A fuzzy variable string length based point symmetry genetic clustering technique is proposed in [13]. It automatically evolves the appropriate types of clusters, both convex and nonconvex, which have some symmetrical structures. It fails if the clusters do not have symmetry property. In [14], a two-stage genetic clustering algorithm (TGCA) is proposed. It can automatically determine the proper number of clusters and the proper partition from a given data set. It is suitable for clustering the data with compact spherical clusters only. Single objective genetic clustering methods [15] fail to solve the issues of clusters shape and size simultaneously. They are suffering from ineffective genetic search, which in turn get stuck at suboptimal clustering solutions [16]. To overcome the limitations of these algorithms some attempts have been made to use multiobjective genetic algorithms. Handl and Knowles [17] proposed multiobjective clustering with automatic K-determination (MOCK) to detect the optimal number of clusters from data sets. But due to the heuristic nature of the algorithm, it provides an approximation to the real (unknown) Pareto front only. Therefore, generation of the best clustering solution cannot be guaranteed, and result shows some variation between individual runs. Saha and Bandyopadhyay [18] proposed a multiobjective clustering technique. In this algorithm points are assigned to different clusters based on the point symmetry based distance. It is able to detect clusters having point symmetry property. However it will fail for clusters having nonsymmetrical shape. Some researchers have applied Differential Evolution (DE) to the task of automatic clustering from data sets. In [19] a Differential Evolution (DE) technique on the point symmetry based cluster validity index is presented. To find the optimal number of clusters, they proposed a modified symmetry based index. The main limitation of this algorithm is problem-dependent dynamic control factor. Suresh et al. [20, 21] applied the Differential Evolution (DE) to the task of automatic fuzzy clustering, where two conflicting fuzzy validity indices are simultaneously optimized. They used a real-coded representation of the search variables to accommodate variable number of cluster centers [20, 21]. It depends on cluster centroid and thus is biased in any sense towards spherical clusters. Tremendous research effort has gone in the past few years to evolve the clusters in complex data sets through evolutionary techniques. Most clustering algorithms assume the number of clusters to be known a priori. The desired granularity [22] is generally determined by external, problem criteria. There seems to be no definite answer to how many clusters are in data set a user defined criterion for the resolution has to be given instead. Second, most of the existing clustering algorithms adopt 2-norm distance measures in the clustering. These measures fail when clusters tend to develop along principal axes. The symmetry based clustering techniques also seek for clusters which are symmetric with respect to their centers. Thus, these techniques will fail if the clusters do not have this property.

The objective of this paper is twofold. First, it aims at the automatic determination of the optimal number of clusters in any data set. Second, it attempts to find clusters of arbitrary shapes and sizes. We show that genetic algorithm with a new line symmetry based distance can give very promising results if applied to the automatic clustering problem. The proposed algorithm evolves near-optimal clustering solutions using multiple clustering criteria, without a priori knowledge of the actual number of clusters. The multiple randomized *Kd* trees based nearest neighbor search is used to reduce the complexity of finding the closest symmetrical points. We refer to this new algorithm as the multiobjective line symmetry based genetic clustering (MOLGC) algorithm. We have compared the MOLGC with two other clustering techniques: SBKM and MOCK. The following performance metrics have been used in the comparative analysis: (1) the accuracy of final clustering results; (2) the computation time; and (3) the statistical significance test.

This paper is organized as follows: The related work on symmetry is reviewed in Section 2. In Section 3, proposed line symmetry measure with multiple randomized *Kd* trees based nearest neighbor search approach is presented. Multiobjective line symmetry based genetic clustering technique is explained in Section 4. Section 5 contains data description and experimental results. Finally, we conclude in Section 6.

2. Related Work

In this section at first, point symmetry based distance is described in brief. Then line symmetry based distance is discussed.

2.1. Point Symmetry Based Distance. Symmetry is considered as a preattentive feature that enhances recognition and reconstruction of shapes and objects. Su and Chou [7] presented an efficient point symmetry distance (PSD) measure to help partitioning the data set into the clusters, where each cluster has the point symmetry property. The point symmetry distance measure between the data point x_i, $\{x_i \mid \text{for } 1 \le i \le N\}$, and the data point x_j relative to the cluster centroid c_k, $\{c_k \mid \text{for } 1 \le k \le K\}$, is defined as

$$d_s\left(x_j, c_k\right) = \min_{\forall i \ne j, 1 \le j \le N} \frac{\left\|\left(x_j - c_k\right) + \left(x_i - c_k\right)\right\|}{\left\|\left(x_j - c_k\right)\right\| + \left\|\left(x_i - c_k\right)\right\|}, \quad (1)$$

for $i \ne j$ and $1 \le i \le N$, where $\|\cdot\|$ denotes the 2-norm distance. Note that distance $d_s(x_j, c_k)$ is minimized when the pattern $x_i = (2 \times c_k - x_j)$ exists in the data sets. They have proposed a symmetry based K-means clustering algorithm called SBKM which assigns the patterns to a particular cluster

depending on the symmetry based distance d_s only when d_s is greater than some user specified threshold $\theta = 0.18$. Otherwise, assignment is done according to the Euclidean distance. We can demonstrate how the point symmetry distance (PSD) measure works well for the case of symmetrical intra-clusters. Assume positions of two centroids c_1 and c_2 are $c_1 = (16, 10)$ and $c_2 = (16, 19)$. The x_1, x_2, and x_3 are three data points and their positions are $x_1 = (14, 16)$, $x_2 = (18, 4)$, and $x_3 = (19, 25)$, respectively. The PSD measure between data points x_1 and cluster center c_1 is

$$d_s(x_1, c_1) = \frac{\|(x_1 - c_1) + (x_2 - c_1)\|}{\|(x_1 - c_1)\| + \|(x_2 - c_1)\|} = \frac{0}{\sqrt{40} + \sqrt{40}} = 0,$$
$$d_s(x_1, c_2) = \frac{\|(x_1 - c_2) + (x_3 - c_2)\|}{\|(x_1 - c_2)\| + \|(x_3 - c_2)\|} = \frac{\sqrt{10}}{\sqrt{13} + \sqrt{45}} = 0.30. \quad (2)$$

Because $d_s(x_1, c_1) < d_s(x_1, c_2)$ and $d_s(x_1, c_1)$ is less than the specified threshold (0.18), the data point x_2 is said to be the most symmetrical point of x_1 relative to c_1; thus we have $x_2 = \text{Arg}\, d_s(x_1, c_1)$. Consequently, assigning the data point x_1 to the cluster C_1 is a good decision. But the problems occurring in the point symmetry distance (PSD) measure are (1) lacking the distance difference symmetry property, (2) leading to an unsatisfactory clustering result for the case of symmetrical inter-clusters. In the first problem, the PSD measure favors the far data point when we have more than two symmetrical data points and this may degrade the symmetrical robustness. We can depict this problem as shown in Figure 1.

Let $x_i = (-5, 0)$, $x_j = (7, 0)$, $x_{j+1} = (10, 0)$, and $c_j = (0, 0)$; then find the most symmetry point of x_i relative to x_j and x_{j+1},

$$d_s(x_i, c_j) = \min\left\{\frac{\|(x_i - c_j) + (x_j - c_j)\|}{\|(x_i - c_j)\| + \|(x_j - c_j)\|}, \frac{\|(x_i - c_j) + (x_{j+1} - c_j)\|}{\|(x_i - c_j)\| + \|(x_{j+1} - c_j)\|}\right\}, \quad (3)$$
$$d_s(x_i, c_j) = \min\left\{\frac{2}{12}, \frac{5}{125}\right\} = \min\{0.16, 0.04\}.$$

The data point x_{j+1} is selected as the most symmetrical point of x_i relative to the centroid c_j. This shows that (1) favors the far data point when we have more than two data points and this may corrupt the symmetrical robustness.

In the second problem, if two clusters are symmetrical to each other with respect to the centroid of any third cluster, then PSD measure gives an unsatisfactory clustering result. As presented in Figure 2, the cluster C_1 and the cluster C_3 are symmetrical to the cluster center c_2.

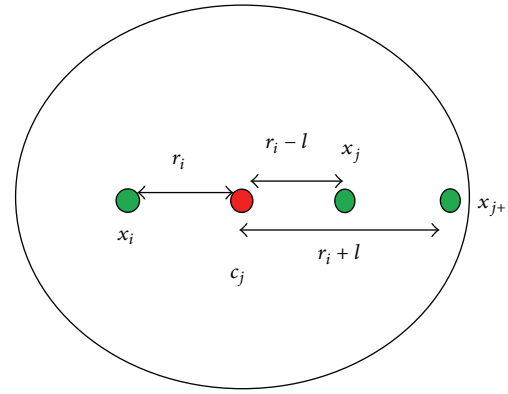


Figure 1: An example for the distance difference symmetry.

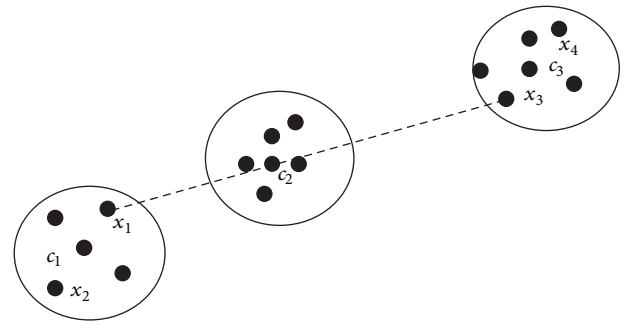


Figure 2: Point symmetry interclusters distance.

Let $x_1 = (-10, -2)$, $x_2 = (-12, -4)$, $x_3 = (12, 2)$, $x_4 = (13, 4)$, $c_1 = (-11, -3)$, $c_2 = (0, 0)$, and $c_3 = (12, 3)$; then for the data point x_1, and by (1), we have

$$d_s(x_1, c_1) = \frac{\|(x_1 - c_1) + (x_2 - c_1)\|}{\|(x_1 - c_1)\| + \|(x_2 - c_1)\|} = \frac{8}{\sqrt{2} + \sqrt{50}} = 0.94,$$
$$d_s(x_1, c_2) = \frac{\|(x_1 - c_2) + (x_3 - c_2)\|}{\|(x_1 - c_2)\| + \|(x_3 - c_2)\|} = \frac{2}{\sqrt{104} + \sqrt{148}} = 0.08, \quad (4)$$
$$d_s(x_1, c_3) = \frac{\|(x_1 - c_3) + (x_4 - c_3)\|}{\|(x_1 - c_3)\| + \|(x_4 - c_2)\|} = \frac{\sqrt{457}}{\sqrt{509} + \sqrt{2}} = 0.89.$$

In the above example point symmetry distance $d_s(x_1, c_2) < d_s(x_1, c_3) < d_s(x_1, c_1)$ is smallest among all distances, so the data point x_1 should be assigned to the cluster C_2, but it conflicts our visual assessment. Due to the above two problems, Chung and Lin [8] proposed a symmetry based distance measure known as Symmetry Similarity Level (SSL), which satisfies the distance difference

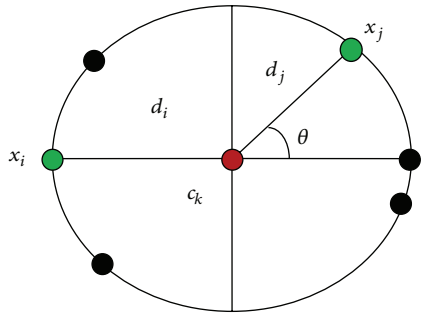


Figure 3: An example for distance difference.

symmetry property. Let c_k denote the cluster centroid; x_i and x_j denote two related data points as shown in Figure 3.

Let $d_i = \overline{x_i c_k}$ and $d_j = \overline{x_j c_k}$; then the Distance Similarity Level (DSL) operator for measuring the distance difference symmetry between the distance $\overline{x_i c_k}$ and the distance $\overline{x_j c_k}$ is defined by

$$\mathrm{DSL}\left(x_i, c_k, x_j\right) = \begin{cases} 1 - \dfrac{\left|d_i - d_j\right|}{d_i}, & \text{if } 0 \le \dfrac{d_j}{d_i} \le 2 \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

They replaced the interval $0 \le d_j/d_i \le 2$ to the interval $0 \le d_j/d_i \le k$, $k > 2$; in (5), the number of examined symmetrical points will be increased and the computational gain might be degraded. They proposed second component called Orientation Similarity Level (OSL). By applying the projection concept, the OSL between the two vectors, $v_i = \overrightarrow{x_i c_k} = (c_k - x_i)$ and $v_j = \overrightarrow{c_k x_j} = (x_j - c_k)$, is defined by

$$\mathrm{OSL}\left(x_i, c_k, x_j\right) = \frac{v_i \cdot v_j}{2\left\|v_i\right\|\left\|v_j\right\|} + 0.5. \tag{6}$$

By (5) and (6), they combined the effect of $\mathrm{DSL}(x_i, c_k, x_j)$ and $\mathrm{OSL}(x_i, c_k, x_j)$ to define a Symmetry Similarity Level (SSL) between the vectors $\overrightarrow{x_i c_k}$ and $\overrightarrow{c_k x_j}$. They defined the following:

$$\mathrm{SSL}\left(x_i, c_k, x_j\right) = \sqrt{\frac{\mathrm{DSL}^2\left(x_i, c_k, x_j\right) + \mathrm{OSL}^2\left(x_i, c_k, x_j\right)}{2}}, \tag{7}$$

for $1 \le k \le K$ and $1 \le i \le N$. Because of $0 \le \mathrm{DSL}(x_i, c_k, x_j) \le 1$ and $0 \le \mathrm{OSL}(x_i, c_k, x_j) \le 1$, it is easy to verify that $0 \le \mathrm{SSL}(x_i, c_k, x_j) \le 1$ is held. Based on SSL operator, Chung and Lin [8] proposed a modified point symmetry based K-means (MPSK) algorithm. The time complexity for finding the symmetry point for n objects is $O(kn^2)$. So this approach is not suitable for large and high dimensional data sets. To overcome limitations of SBKM, Bandyopadhyay and Saha [23] proposed new point symmetry based clustering algorithm known as variable string length genetic clustering technique with point symmetry (VGAPS). The proposed point symmetry distance is defined as follows.

The symmetrical (reflected) point of x with respect to a particular centre c is $2 \times c - x$ that is denoted by x^*. Let k unique nearest neighbors of x^* be at Euclidean distances of d_i, $i = 1, 2, \ldots, k$. Then $d_{\mathrm{ps}}(x, c) = d_{\mathrm{sym}}(x, c) \times d_e(x, c)$:

$$d_{\mathrm{ps}}(x, c) = d_{\mathrm{sym}}(x, c) \times d_e(x, c) = \frac{\sum_{i=1}^{k} d_i}{k} \times d_e(x, c), \tag{8}$$

where $d_e(x, c)$ is the Euclidean distance between the point x and cluster center c. It can be seen from (8) that k cannot be chosen equal to 1, since if point x^* exists in the data set then $d_{\mathrm{ps}}(x, c) = 0$ and hence there will be no impact of the Euclidean distance. To overcome this problem, they have taken average distance between reflected point x^* and its first and the second unique nearest neighbor's points. They proposed a rough guideline of the choice of θ, the threshold value on the point symmetry distance that is equal to maximum nearest neighbor distance $d_{\mathrm{NN}}^{\max}$ in the data set. For reducing the complexity of point symmetry distance computation, *Kd* tree based data structure is used. VGAPS detects clusters which are point symmetry with respect to their centers. Thus VGAPS will fail if the clusters do not have this property.

2.2. Existing Line Symmetry Based Distance. From the geometrical symmetry view point, point symmetry and line symmetry are two widely discussed issues. Motivation by this, Saha and Maulik [24] proposed a new line symmetry based automatic genetic clustering technique called variable string length genetic line symmetry distance based clustering (VGALS-Clustering). To measure amount of line symmetry of a point x with respect to a particular line i, $d_{\mathrm{ls}}(x, i)$, the following steps are followed:

(1) For a particular data point x, calculate the projected point p_i on the relevant symmetrical line i.

(2) Find $d_{\mathrm{sym}}(x, p_i)$ as

$$d_{\mathrm{sym}}(x, p_i) = \frac{\sum_{i=1}^{k} d_i}{k}, \tag{9}$$

where k nearest neighbors of $x^* = 2 \times p_i - x$ are at Euclidean distances of d_i, $i = 1, 2, \ldots, k$. Then the amount of line symmetry of a particular point x with respect to that particular symmetrical line of cluster i is calculated as

$$d_{\mathrm{ls}}(x, i) = d_{\mathrm{sym}}\left(x, p_i\right) \times d_e(x, c), \tag{10}$$

where c is the centroids of the particular cluster i and $d_e(x, c)$ is the Euclidean distance between the points x and c. The possible problem existing in this given line symmetry measure is lacking the closure property. The closure property can be expressed as follows: if the data point p_i is currently assigned to the cluster centroid c_k in the current iteration, the determined most symmetrical point p_j relative to c_k must have been assigned to c_k in the previous iteration.

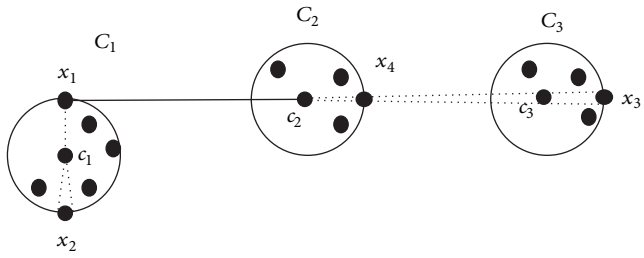


Figure 4: Violation of closure property.

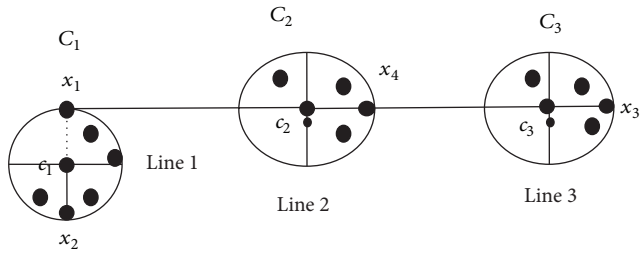


Figure 5: An example of proposed line symmetry distance.

3. Proposed Line Symmetry Measure

Both point symmetry and line symmetry distance lack the closure property and this would result in an unsatisfactory clustering result. According to the symmetry property, the data point x_1 in Figure 4, which is not in the cluster C_2 originally, if symmetry distance $d_{\text{sym}}(x_1, c_2)$ of point x_1 with cluster center c_2 is the most symmetrical distance ($d_{\text{sym}}(x_1, c_2) < d_{\text{sym}}(x_1, c_1) < d_{\text{sym}}(x_1, c_3)$) among other symmetry distances, it tells us that data point x_1 should currently be assigned to the cluster C_2. But the most symmetrical point of x_1 relative to the centroid c_2 is the data point x_3, which has been assigned to the centroid c_3. Since the data point x_3 has not been assigned to the centroid c_2 before, it violates closure property. It would give an unsatisfactory clustering result.

By considering above problem in existing symmetry based distances, we have applied a constraint in new line symmetry distance measure to satisfy closure property, in which, to compute the line symmetry distance of the data point x_i, we have restricted the candidate symmetrical points $x_j \notin C_k$ relative to each symmetrical line k of the corresponding cluster C_k. For the data point x_i relative to symmetrical line of cluster C_k, this restriction can help us to search more suitable symmetrical point x_j, because we ignore the candidate most symmetrical point x_j which is not in the cluster C_k. As depicted in Figure 5, let the point x_1 have most line symmetry distance ($d_{\text{ls}}(x_1, C_2) < d_{\text{ls}}(x_1, C_1) < d_{\text{ls}}(x_1, C_3)$) with respect to particular line of cluster C_2 and the symmetrical point is x_3, but due to the above constraint the proposed line symmetry distance method is assigned the point x_1 to cluster C_1. The assignment of x_1 to the cluster C_1 is a reasonable assignment from our visual system. We applied the second modification in which the first and second symmetrical points ${x_1}^*$ and ${x_2}^*$ of point x_i are found in cluster C_k (as shown in Figure 6)

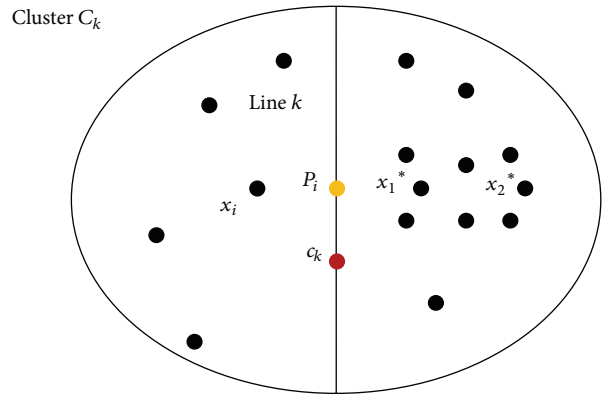


Figure 6: An example of two symmetry points.

relative to the symmetrical line, not in all data points; that is, each point x_i, $1 \le i \le n$, is assigned to cluster C_k iff $d_{\text{ls}}(x_i, C_k) \le d_{\text{ls}}(x_i, C_j)$, where $j, k = 1, \ldots, k$ and $j \ne k$, $d_{\text{ls}}(x_i, C_k)/d_e(x_i, c_k) \le \theta$, and ${x_1}^*$ and ${x_2}^*$ belong to cluster C_k. The distance $d_{\text{ls}}(x_i, C_k)$ is calculated as given in (22) and $\theta = d_{\text{nn}}^{\max}$ is the symmetrical threshold, where $d_{\text{nn}}^{\max} = \max_{i=1,\ldots,n} d_{\text{nn}}(x_i)$ and the distance $d_{\text{nn}}(x_i)$ is the maximum nearest neighbor distance in the data set.

The value of θ is kept equal to the maximum nearest neighbor distance among all the points in the data set. Point assignment based on proposed line symmetry distance is given in Algorithm 1.

For computing the proposed line symmetry distance of a given data set, we find the symmetrical line of each cluster by using central moment method [25] that is used to measure the Symmetry Similarity Level between two data points relative to symmetrical line. Let the data set be denoted by $X = \{(x_1, y_1), (x_2, y_2), (x_n, y_n)\}$; then the (p, q)th order moment is defined as

$$m_{pq} = \sum_{\forall (x_i, y_i) \in X} x_i^p y_i^q. \tag{11}$$

The centroid of the given data set for one cluster is defined as $(m_{10}/m_{00}, m_{01}/m_{00})$. The central moment is defined as

$$u_{pq} = \sum_{\forall (x_i, y_i) \in X} (x_i - \overline{x})^p (y_i - \overline{y})^q, \tag{12}$$

where $\overline{x} = m_{10}/m_{00}$ and $\overline{y} = m_{01}/m_{00}$. According to the calculated centroid and (12), the major axis of each cluster can be determined by the following two items:

(a) The major axis of the cluster must pass through the centroids.

(b) The angle between the major axis and the x-axis is equal to $(1/2)\tan^{-1}(2u_{11}/(u_{20} - u_{02}))$.

Consequently, for one cluster, its corresponding major axis is thus expressed by

$$\left(\left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right), \frac{1}{2} \tan^{-1} \left(\frac{2u_{11}}{u_{20} - u_{02}} \right) \right). \tag{13}$$

Procedure: Point assignment based on proposed line symmetry distance( )
for $i = 1$ to n do
  for $k = 1$ to K do
    Find the first and the second symmetrical points $x_1{}^*$ and $x_2{}^*$ of x_i relative to a projected point p_i on line k of cluster C_k/* *To ensure the closure property* */
    Calculate the line symmetry-based distance $d_{\text{ls}}[C_k] = d_{\text{ls}}(x_i, C_k), k = 1, 2, \ldots, K$
  end for
  Find $C_k^* = \text{Arg}\min_{k=1,\ldots,K} d_{\text{ls}}[C_k]$
  if $d_{\text{ls}}(x_i, C_k^*) \le d_{\text{ls}}(x_i, C_l)$, where $k, l = 1, \ldots, K$ and $k \ne l, d_{\text{ls}}(x_i, C_k^*)/d_e(x_i, c_k^*) \le \theta$ then
    Assign the point x_i to the cluster C_k^*
    Else
    Assign the point x_i to the cluster $C_k{}^*$ based on the Euclidean distance measure, $C_k^* = \text{Arg}\min_{k=1,\ldots,K} d_e(x_i, c_k^*)$
  end if
end for
Compute new cluster centers of the K clusters as follows:

$$c_k^{\text{new}} = \frac{1}{n_k} \sum_{i \in S_k} x^i,$$

where n_k is the number of data points belonging to the cluster C_k and S_k is set of data points which have been assigned to the cluster center c_k.

Algorithm 1: Point assignment based on proposed line symmetry distance.

Let normalized form of data points be stored into memory of the computer system. Now we can apply central moment method for computing the shape of the data points. A brief mathematical calculation for finding the symmetrical line of each cluster by using central moment method is given in Figure 7 and below:

$$\begin{aligned}
m_{00} &= \sum_{\forall(x_i,y_i)\in X} x_i^0 y_i^0 = \text{Area} = 11,\\
m_{01} &= \sum_{\forall(x_i,y_i)\in X} x_i^0 y_i^1 = 44,\\
m_{10} &= \sum_{\forall(x_i,y_i)\in X} x_i^1 y_i^0 = 44,\\
m_{11} &= \sum_{\forall(x_i,y_i)\in X} x_i^1 y_i^1 = 178,\\
m_{02} &= \sum_{\forall(x_i,y_i)\in X} x_i^0 y_i^2 = 190,\\
m_{20} &= \sum_{\forall(x_i,y_i)\in X} x_i^2 y_i^0 = 190.
\end{aligned} \tag{14}$$

The centroid of cluster is calculated as

$$\overline{x} = \frac{m_{10}}{m_{00}} = \frac{44}{11} = 4, \qquad \overline{y} = \frac{m_{01}}{m_{00}} = \frac{44}{11} = 4. \tag{15}$$

We can apply centroid $(\overline{x}, \overline{y}) = (4, 4)$ of cluster for computing the central moments. The physical significance of the central moments is that they just give the area and the moment of inertia. The lower order central moments (Zero_{th}) give the area of the region R:

$$u_{00} = \sum_{\forall(x_i,y_i)\in X} (x_i - 4)^0 (y_i - 4)^0 = 11. \tag{16}$$

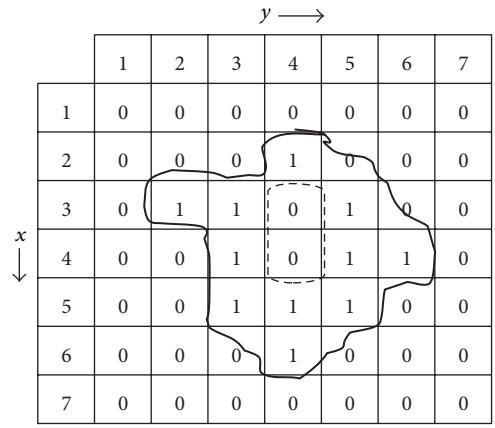


Figure 7: A region R in binary image.

The product moment involves finding the product of $(x_i - \overline{x})$ and $(y_i - \overline{y})$ increasing to a power

$$u_{11} = \sum_{\forall(x_i,y_i)\in X} (x_i - 4)^1 (y_i - 4)^1 = 2. \tag{17}$$

The second order central moment along x-axis is

$$u_{02} = \sum_{\forall(x_i,y_i)\in X} (x_i - 4)^0 (y_i - 4)^2 = 14. \tag{18}$$

The second order central moment along y-axis is

$$u_{20} = \sum_{\forall(x_i,y_i)\in X} (x_i - 4)^2 (y_i - 4)^0 = 13. \tag{19}$$

The angle between the major axis and the x-axis is

$$\frac{1}{2}\tan^{-1}\left(\frac{2u_{11}}{u_{20} - u_{02}}\right) = 38^o. \tag{20}$$

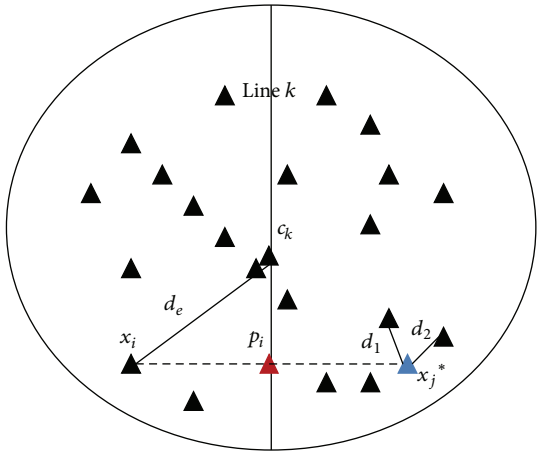


Figure 8: An example for computing line symmetry distance.

The obtained major axis is treated as the symmetric line of the relevant cluster. This symmetrical line is used to measure the amount of line symmetry of a particular point in that cluster. In order to measure the amount of line symmetry of a point (x_i) with respect to a particular line k of cluster C_k, $d_{\text{ls}}(x_i, C_k)$, the following steps are followed.

(1) For a particular data point x_i, calculate the projected point p_i on the relevant symmetrical line k of cluster C_k (as shown in Figure 8) and then find out all possible symmetrical data point x_j relative to each symmetrical line k for $1 \le i \le n, 1 \le j \le n$, and $1 \le k \le K$.

(2) Find $d_{\text{sym}}(x_i, p_i)$ as

$$d_{\text{sym}}(x_i, p_i) = \frac{\sum_{i=1}^{knear} d_i}{knear}, \tag{21}$$

where k nearest neighbors of $x_j = 2 \times p_i - x_i$ are at Euclidean distances of d_i, $i = 1, 2, \ldots, knear$. In fact, the role of the parameter $knear$ is intuitively easy to understand and it can be set by the user based on specific knowledge of the application. In general, a fixed value of $knear$ may have many drawbacks. For clusters with too few points, the points likely to be scattered and the distance between two neighbors may be too large. For very large cluster fixed number of neighbors may not be enough because few neighbors would have a distance close to zero. Obviously, the parameter $knear$ is related to the expected minimum cluster size and should be much smaller than the number of objects in the data. To gain a clear idea of the distance of the neighborhood of a point, we have chosen $knear \le \sqrt{n}$ in our implementation. The randomized Kd trees based nearest neighbor search is used to reduce the computation time of the nearest neighbors. The amount of line symmetry of a particular point x_i with respect to particular symmetrical line of cluster C_k is calculated as

$$d_{\text{ls}}(x_i, C_k) = d_{\text{sym}}(x_i, p_i) \times d_e(x_i, c_k), \tag{22}$$

where c_k is the centroid of the cluster C_k and $d_e(x_i, c_k)$ denotes Euclidean distance between data point x_i and cluster center c_k.

3.1. Multiple Randomized Kd Trees Based Nearest Neighbor Search. The problem of nearest neighbor search is one of major importance in a variety of applications such as image recognition, data compression, pattern recognition and classification, machine learning, document retrieval systems, statistics, and data analysis. The most widely used algorithm for nearest neighbor search is the K dimensional tree (Kd tree) [26–30]. This works well for exact nearest neighbor search in low dimensional data but quickly loses its effectiveness as dimensionality increases. In high dimensions to find the nearest neighbor may require searching a very large number of nodes. However, solving this problem in high dimensional spaces seems to be a very difficult task and there is no algorithm that performs significantly better than the standard brute-force search. To address this problem, Anan and Hartley [31] have investigated the following strategies: (1) They create m different Kd trees each with a different structure in such a way that searches in the different trees will be (largely) independent. (2) With a limit of n nodes to be searched, they break the search into simultaneous searches among all the m trees. On average, n/m nodes will be searched in each of the trees. (3) The principal component analysis is used to rotate the data to align its moment axes with the coordinate axes. Data will then be split up in the tree by hyperplanes perpendicular to the principal axes. They have written that either by using multiple search trees or by building the Kd tree from data realigned according to its principal axes, search performance improves and even improves further when both techniques are used together.

To overcome the above problem, we have used the approximate nearest neighbor search approach, in which the randomized trees are built by choosing the split dimension randomly from the first d dimensions on which data has the greatest variance and each tree is constructed independently [32]. In proposed MOLGC algorithm, instead of always splitting on the maximally variant dimension, each tree selects randomly among the top five most variant dimensions at each level. When searching the trees, a single priority queue is maintained across all the randomized trees so that search can be ordered by increasing distance to each bin boundary. The degree of approximation is determined by examining a fixed number of leaf nodes, at which point the search is terminated and the best candidates returned. In the multiple randomized Kd trees based nearest neighbor search technique, the data points $X = x_1, x_2, \ldots, x_n$ are preprocessed into a metric space S, so that, given any query point $q \in X$, the nearest or generally k nearest points of x to q can be reported efficiently. In proposed MOLGC algorithm to find line symmetric distance of a particular point x_i with respect to the symmetrical line of cluster C_k, we have to find the nearest neighbors of x_j (where $x_j = 2 \times p_i - x_i$ for $1 \le i \le n$ and $1 \le j \le n$). Therefore the query point q is set equal to x_j. After getting the k nearest neighbors of x_j, the line symmetrical distance of x_i to the symmetrical line of cluster C_k is calculated by using (22).

```
Begin
Generate the initial population P/* Popsize = |P|*/
while (the termination criterion is not satisfied)
    for i = 1 to P
        Assign the points based on line symmetry based technique
        Call Procedure: Point assignment based on proposed line symmetry distance() to compute
        the fitness of the chromosome
        Compute objective functions for current chromosomes
        select the chromosomes to apply genetic operators
        Apply crossover operation with probability p_c
        Apply mutation operator to the selected chromosome with mutation probability p_m
        Compute objective functions for new offsprings
    end for
end while
Select the best solution from population
End
```

Algorithm 2: Steps of proposed algorithm.

4. Multiobjective Line Symmetry Based Genetic Clustering Technique

In this section, a multiobjective genetic clustering technique using the proposed line symmetry based distance is proposed. The algorithm is known as multiobjective line symmetry based genetic clustering (MOLGC). The subsections of this section are organized as follows.

4.1. Chromosome Representation. In proposed algorithm, the numerical feature values of all cluster centers are encoded into a real coded chromosome as a clustering solution. The length of a particular chromosome c is l_c, given by $l_c = d \times K$, where d is the dimension of the data set and K is the number of cluster centers encoded in that chromosome.

For example a chromosome representation (5.5 3.5 4.2 4.0 2.5 3.5 6.2 7.3 1.5 2.5 3.5 4.5) has three cluster centers in four dimensional feature space. The encoded three clusters are (5.5 3.5 4.2 4.0), (2.5 3.5 6.2 7.3), and (1.5 2.5 3.5 4.5). For a variable-length chromosome representation, each chromosome has the initial length l_c. The number of clusters, denoted by K, is randomly generated in the range $[K_{\min}, K_{\max}]$. Here $K_{\min}$ is chosen as 2, and $K_{\max}$ is chosen to be $\sqrt{n}$, where n is the size of the data set. Their after K-means algorithm is executed with the set of centers encoded in each chromosome. The resultant centers are used for replacing the centers in the corresponding chromosomes. The steps of proposed MOLGC algorithm are given in Algorithm 2.

4.2. Fitness Computation. The fitness of an individual indicates the degree of suitability of the solution it represents. In general, the fitness of a chromosome is evaluated using the objective function of the problem. The first objective of the clustering problem considered in this paper is to maximize the similarity within each cluster and the dissimilarity among clusters. The second objective is to detect clusters based on line symmetry distance. In this paper, two objective functions, the Davies-Bouldin (DB) index [33] and proposed line symmetry distance, are used to evaluate the fitness of a chromosome. The DB index is used to find clusters which are compact and well separated by minimizing the intracluster distance while maximizing the intercluster distance. DB index is the ratio of the sum of within cluster scatter $S_{i,q}$ of cluster C_i to between cluster separations. Within cluster scatter $S_{i,q}$ of cluster C_i is defined as

$$S_{i,q} = \left(\frac{1}{C_i} \sum_{x \in C_i} \left\| x - c_i \right\|^{q/2} \right)^{1/q}, \tag{23}$$

where c_i denotes the cluster center of cluster C_i. Cluster center c_i is computed as

$$c_i = \frac{1}{n_i} \sum_{x \in C_i} x, \tag{24}$$

where n_i denotes the number of the objects belonging to cluster C_i. The within cluster scatter $S_{i,q}$ denotes the qth root of the qth moment of the objects belonging to cluster C_i with respect to their mean. The distance between clusters C_i and C_j is denoted as d_{ij} and is defined as $d_{ij} = d_e(c_i, c_j)$, where d_e stands for Euclidean distance between the centroids c_i and c_j of the clusters C_i and C_j, respectively. Then, DB index is defined as

$$\mathrm{DB} = \frac{1}{k} \sum_{i=1}^{k} R_{i,q}. \tag{25}$$

Here,

$$R_{i,q} = \max_{i, i \neq j} \left\{ \frac{S_{i,q} + S_{j,q}}{d_{ij}} \right\}, \tag{26}$$

where k corresponds to the number of selected clusters. An individual cluster index is taken as the maximum pairwise comparison computed as the ratio of the sum of within cluster

```
Procedure: Fitness( ) //Fitness computation of chromosome
Assign the data points based on the procedure line symmetry
f = 0
for k = 1 to K do
   for each data point x_i, i = 1, ..., n and x_i ∈ C_k do
      f = f + d_ls(x_i, C_k)
   end for
end for
```

Algorithm 3: The procedure of the fitness computation.

dispersions from the two partitions divided by a measure of the between cluster separation. Smaller values for DB index correspond to good clusters. We set the fitness F_i of chromosome i to be equal to $1/\mathrm{DB}_i$, where DB_i is the DB index of individual i. The second objective function is based on proposed line symmetry distance. The procedure of the fitness computation is given in Algorithm 3.

The fitness function of the chromosomes $\mathrm{fit}_{\mathrm{ls}}$ is defined as the inverse of f; that is,

$$\mathrm{fit}_{\mathrm{ls}} = \frac{1}{f}. \tag{27}$$

4.3. Genetic Operators. In this subsection, genetic operators used in proposed clustering algorithm are discussed. These genetic operators pass genetic information between subsequent generations of the population.

4.3.1. Selection. Pareto based selection is used to select fitter solutions in each step of the evolution. It is a stochastic selection method where the selection probability of a chromosome is proportional to the value of its fitness [34]. The fitness for a chromosome chrom, denoted by fitness (chrom), is converted from its Pareto count (or dominance count) of x in the whole population. The probability that the individual chrom is selected from the population is denoted by

$$p_r(i) = \frac{\text{fitness (chrom)}}{\sum_{i=1}^{P} \text{chrom}_i}, \tag{28}$$

where P is the population size; comparing with the conventional roulette wheel selection method that is directly based on the fitness of solutions, Pareto-dominance based selection method can lower the selection pressure and increase the chances of the subspaces with low fitness to be selected into next population.

4.3.2. Crossover. Crossover is a probabilistic process that exchanges information between two parent chromosomes for generating two child chromosomes [34]. For chromosomes of length l_c, a random integer, called the crossover point, is generated in the range $[1, l_c - 1]$. The portions of the chromosomes lying to the right of the crossover point are exchanged to produce two offspring. Let parent chromosomes C_1 and C_2 encode k_1 and k_2 cluster centers, respectively. l_1, the crossover point in C_1, is generated as $l_1 = \text{rand}() \bmod k_1$. Let l_2 be the crossover point in C_2, and it may vary in between $[\mathrm{LB}(k_2), \mathrm{RB}(k_2)]$, where LB() and RB() indicate the left and right bounds of the range of k_2, respectively. $\mathrm{LB}(k_2)$ and $\mathrm{RB}(k_2)$ are given by

$$\begin{aligned} \mathrm{LB}(k_2) &= \min[2, \max[0, 2 - (k_1 - l_1)]], \\ \mathrm{RB}(k_2) &= [k_2 - \max[0, 2 - l_1]]. \end{aligned} \tag{29}$$

Therefore l_2 is given by

$$l_2 = \begin{cases} \mathrm{LB}(l_2) + \text{rand}() \bmod (\mathrm{RB}(l_2) - \mathrm{LB}(l_2)), \\ \qquad \text{if } \mathrm{RB}(l_2) \geq \mathrm{LB}(l_2) \\ 0 \qquad \text{otherwise.} \end{cases} \tag{30}$$

As an example, let two chromosomes C_1 (10 20 15 25) and C_2 (15 30 18 32 19 35 25 45 30 50) be with number of 2 and 5 clusters. Now we can apply crossover operation on C_1 and C_2 as

$$\begin{aligned} C_1 &= (10\,20 \;\; 15\,25), \\ C_2 &= (15\,30 \;\; 18\,32 \;\; 19\,35 \;\; 25\,45 \;\; 30\,50). \end{aligned} \tag{31}$$

The crossover point in C_1 is generated as

$$l_1 = 5 \bmod 2 = 1, \tag{32}$$

where 5 is random number generated by rand() function. The crossover point in C_2 varies in between $\mathrm{LB}(l_2) = \min[2, \max[0, 2 - (2 - 1)]] = 1$ and $\mathrm{RB}(l_2) = [5 - \max[0, 2 - 1]] = 4$.

The crossover point in C_2 is $l_2 = \mathrm{LB}(l_2) + \text{rand}() \bmod (\mathrm{RB}(l_2) - \mathrm{LB}(l_2)) = 1 + 5 \bmod (4 - 1) = 1 + 2 = 3$

$$\begin{aligned} & \qquad\qquad\quad l_1 \\ C_1 &\;\; (10\,20 \mid 15\,25) \qquad l_2 \\ C_2 &\;\; (15\,30 \;\; 18\,32 \;\; 19\,35 \mid 25\,45 \;\; 30\,50). \end{aligned} \tag{33}$$

The offspring C_3 and C_4 generated after crossover operation are

$$\begin{aligned} &\text{Offspring } C_3 \;\; (10\,20 \;\; 25\,45 \;\; 30\,50), \\ &\text{Offspring } C_4 \;\; (15\,30 \;\; 18\,32 \;\; 19\,35 \;\; 15\,25). \end{aligned} \tag{34}$$

Crossover probability is selected adaptively as in [35]. Let $f_{\max}$ be the maximum fitness value of the current population,

$\overline{f}$ the average fitness value of the population, and f' the larger of the fitness values of the solutions to be crossed. Then the probability of crossover, p_c, is calculated as

$$p_c = k_1 \times \frac{(f_{\max} - f')}{(f_{\max} - \overline{f})}, \quad \text{if } f' > \overline{f},$$
$$p_c = k_3, \quad \text{if } f' \leq \overline{f}. \tag{35}$$

Here, the values of k_1 and k_3 are equal to 1.0 [35]. Clearly, when $f_{\max} = \overline{f}$ then $f' = f_{\max}$ and p_c will be equal to k_3. The value of p_c increases when the chromosome is quite poor. In contrast if p_c is low it means chromosome is good. It will prevent the proposed MOLGC algorithm from getting stuck at local optimum.

4.3.3. Mutation. Each cluster center in a chromosome is changed with a random variable generated from a Laplacian distribution [18]. This distribution is characterized by location μ (any real number) and scale δ parameters. The probability density function of Laplace (a, b) is

$$p(x) = \frac{1}{2b} e^{-|x-a|/b}, \tag{36}$$

where the scaling parameter b sets the magnitude of perturbation that is referred to as the diversity. Here, parameter a denotes the location value which is to be perturbed. We set the scaling parameter b equal to 1.0 in our experimental results. In this mutation operation the old value at the mutation position of a chromosome is replaced with newly generated random value using Laplace distribution. The mutation operation is applied for all dimensions of data set independently. The mutation probability p_m is selected adaptively for each chromosome as in [35]. The expression is given below:

$$p_m = k_2 \times \frac{(f_{\max} - f)}{(f_{\max} - \overline{f})}, \quad \text{if } f > \overline{f},$$
$$p_m = k_4, \quad \text{if } f \leq \overline{f}, \tag{37}$$

where k_2 and k_4 are equal to process 0.5. The adaptive mutation process assists genetic algorithm to come out of local optimum. When $f_{\max} = \overline{f}$ value decreases then p_c and p_m both will be increased. As a result GA will come out of local optimum. It will also happen for the global optimum and may result in interference of the near optimal solutions. As a result genetic algorithm never converges to the global optimum. But the values of adaptive crossover probability p_c and adaptive mutation probability p_m will be higher for low fitness solutions and will get low values for higher fitness solutions. The high fitness solutions aid in convergence of the genetic algorithm and the low fitness solutions prevent the genetic algorithm from getting stuck at a local optimum. It may be possible for a solution with highest fitness value; p_c and p_m are both 0. As a result the best solution is transferred into the next generation without crossover and mutation. For selection operator this may lead to an exponential growth of the solution in the population and may cause premature convergence. To overcome the above problem, a default mutation rate (of 0.01) is kept for every solution in the proposed algorithm MOLGC.

4.4. Termination Criterion. The proposed multiobjective clustering algorithm has been executed for a fixed number of generations. The fixed number is supplied by the user for terminating the algorithm. After termination, the algorithm gives the best string of the last generation that provides the solution to the clustering problem.

5. Experimental Evaluation

The experiments reported in this section were performed on a 2.0 GHz Core 2 Duo processors with 2 GB of memory. We have tested proposed MOLGC algorithm on both real and synthetic data. The qualities of clustering results are measured by adjusted Rand index. We compared the performance of SBKM, MOCK, and MOLGC algorithms. The source code of SBKM is available on Ref. http://mail.tku.edu.tw/chchou/others/SBKM.rar. The source code for the MOCK algorithm is obtained from Ref. (http://personalpages.manchester.ac.uk/mbs/julia.handl/mock.html). For the purpose of comparison, another multiobjective clustering technique, MOCK, is also executed on the above mentioned data sets with default parameter settings. In order to show the effectiveness of the proposed MOLGC clustering technique over existing symmetry based clustering techniques, a symmetry based K-means (SBKM) algorithm is executed on both real and synthetic data.

5.1. Parameter Setting. The proper setting of parameters in genetic algorithm is crucial for its good performance. Different parameter values might yield very different results. A good setting for algorithm may give best solution within a reasonable time period. In contrast, a poor setting might cause the algorithm to be executed for a very long time before finding a good solution. Sometimes it may so happen that it is not able to find a good solution at all. Grefenstette [36] has used genetic algorithm to investigate the optimal parameters of genetic algorithms. He has reported the best parameter values for GA; these are population size = 30, number of generations = not specified, crossover rate of 0.9, and mutation rate of 0.01. However, the selection of optimal parameters in GA is domain dependent and relies on the specific application areas. Below we justify how the used parameters are selected in MOLGC.

(1) Population size: Goldberg [37] has theoretically analyzed that the optimal population size increases exponentially and is rather large for even moderate chromosome lengths. It has been shown in [37] that the number of schemata processed effectively is proportional to n^3, where n is the population size. This seems to justify the selection of large population size. However, the larger the population size, the longer the genetic algorithm takes to compute each generation. Motivated by above discussion, we set population size = 50 in our proposed algorithm (MOLGC).

(2) Number of generations: A GA generally converges within a few generations. The pure selection convergence times are $O(\log N)$ generations, where N is the size of the population. Thus GA generally searches fairly quickly. In [37] it is mentioned that for a given adequate population size if some linkage knowledge is incorporated into the chromosomes then it is expected that mixing of good building blocks can take place before convergence. Thus it is important to detect near-uniformity of the population and terminate the GA, before wasting function evaluations on an inefficient, mutation-based search. So we set number of generations = 50 (executing MOLGC further did not improve its performance).

(3) Initialization of population: It is customary to initialize genetic algorithm with a population of random individuals. But sometimes previously known (good) solutions can be used to initialize a fraction of the population and this results in faster convergence of GA. In the proposed MOLGC, after randomly generating the cluster centers, some iterations of K-means algorithm are executed to separate the cluster centers as much as possible.

(4) Selection of crossover and mutation probabilities: These are two basic parameters of GA. The crossover operation is performed based on crossover probability (μ_c). If $\mu_c = 0$, then child offspring is the same copy of parents. If $\mu_c > 0$, then offspring is result of crossover operation on parents chromosome. If $\mu_c = 1$, then all offspring are made by crossover. Crossover operation is performed so that good fitness value parent chromosomes can be combined in the offspring to result in potentially improved solutions. However, it is good to leave some parts of population to survive for the next generation. Mutation probability (μ_m) determines how often parts of a chromosome are mutated. If there is no mutation, offspring is taken after crossover (or copy) without any change. If mutation is performed (i.e., $\mu_m > 0$), a part of a chromosome is changed. If mutation probability is 100%, the whole chromosome is changed; if it is 0%, nothing is changed. Mutation is made to prevent GA from falling into local optima. But it should not occur very often; otherwise GA will change to random search. In MOLGC initially the mutation probability and crossover probability were kept fixed. We obtained good results with combination of $\mu_c = 0.8$ and $\mu_m = 0.01$.

The parameters used for proposed MOLGC algorithm in our experimental study are given in Table 1. Apart from the maximum number of clusters, these parameters are kept constant over the entire range of data sets in our comparison study. In this comparison study, the SBKM algorithm is executed for 50 iterations. The parameter θ is chosen equal to 0.18 for all data sets. For MOCK algorithm the total number of generation is kept equal to 50.

Table 1: Parameter setting for proposed MOLGC algorithm.

Parameter	Setting
Number of generations	50
Population size	50
Number of clusters	$K_{\min}$ to $K_{\max}$ [2 to 20]
Crossover probability (p_c)	0.8
Mutation probability (p_m)	0.01

In order to evaluate the performance of the proposed multiobjective genetic clustering algorithm more objectively, eight artificial data sets and three real data sets are used.

5.2. Artificial Data Sets. The artificial data set-1, data set-2, data set-3, and data set-4 are obtained from [7, 17] and remaining data sets were generated by two data generators (http://personalpages.manchester.ac.uk/mbs/julia.handl/generators.html). These generators permit controlling the size and structure of the generated data sets through parameters, such as number of points and dimensionality of the data set.

(1) Data set-1: This data set, used in [7], contains 300 points distributed on two crossed ellipsoidal shells. This is shown in Figure 9(a).

(2) Data set-2: This data set, used in [7], is combination of ring shaped, compact, and linear clusters. The total number of points in it is 300. The dimension of this data set is two. This is shown in Figure 9(b).

(3) Data set-3: This data set, used in [7], consists of 250 data points distributed over five spherically shaped clusters. This is shown in Figure 9(c).

(4) Data set-4: This data set, used in [17], consists of 1000 data points distributed over four square clusters. This is shown in Figure 9(d).

(5) Data set-5: This data set contains 10 dimensional 838 data point's distributed over Gaussian shaped four clusters.

(6) Data set-6: This data set consists of 10 dimensional 3050 data points distributed over Gaussian shaped ten clusters.

(7) Data set-7: This data set is a 50 dimensional data set and it consists of 351 data points distributed over ellipsoid shaped four clusters.

(8) Data set-8: This data set contains 50 dimensional 2328 data points distributed over ellipsoid shaped ten clusters.

The real data sets are obtained from UCI repository (http://archive.ics.uci.edu/ml/). For experimental results four real data sets are considered.

(1) Iris: Iris data set consists of 150 data points distributed over three clusters. Each cluster has 50 points. This data set represents different categories of irises characterized by four feature values. It has three classes,

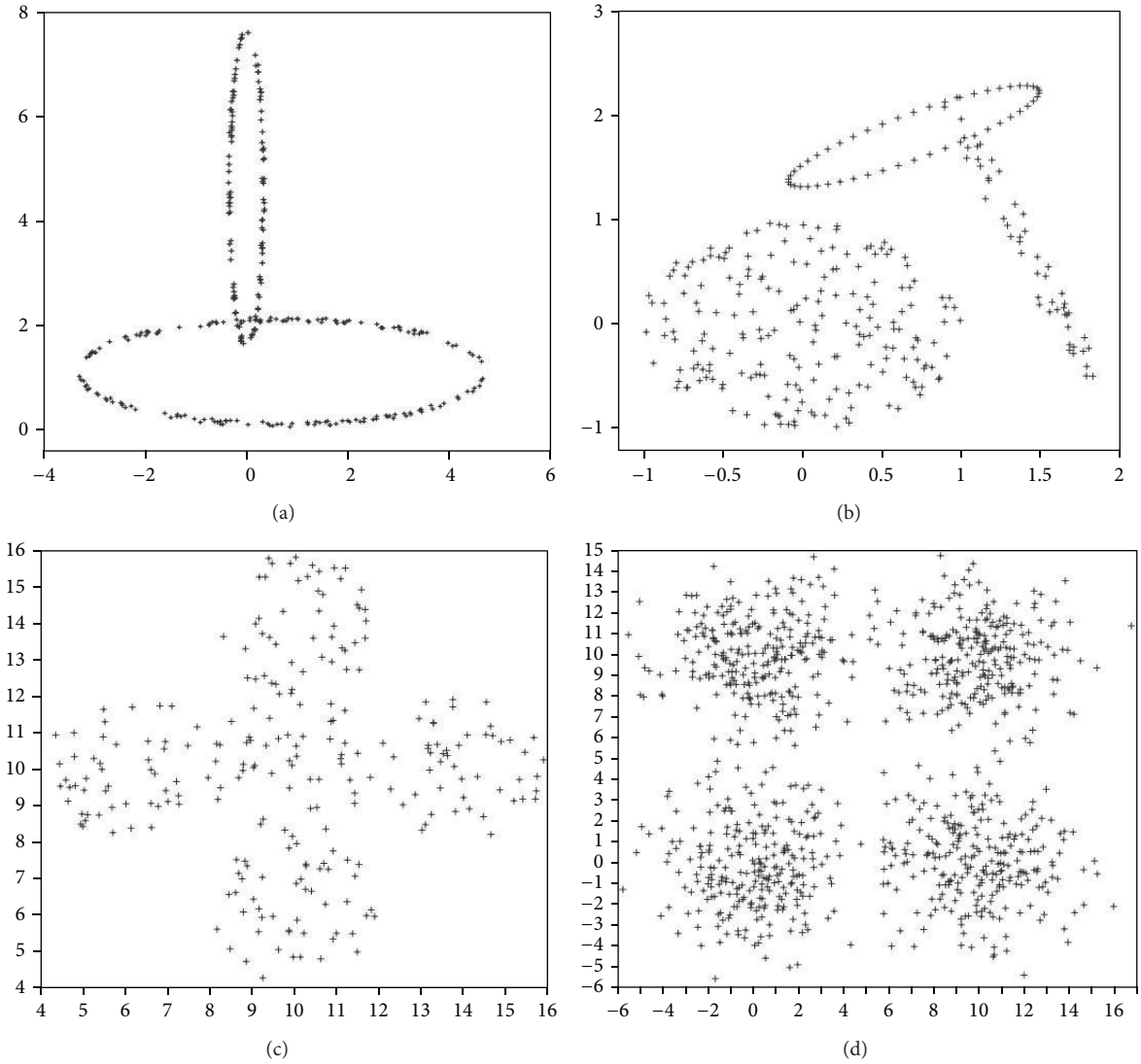


Figure 9: (a) The data set containing two crossed ellipsoidal shells. (b) The data set containing ring shaped, compact, and linear clusters. (c) The data set containing five spherically shaped clusters. (d) The data set containing four linear clusters.

Setosa, Versicolor, and Virginica, among which the last two classes have a large amount of overlap while the first class is linearly separable. The sepal area is obtained by multiplying the sepal length by the sepal width and the petal area is calculated in an analogous way.

(2) Cancer: Wisconsin breast cancer data set consists of 683 sample points. Each pattern has nine features corresponding to clump thickness, cell size uniformity, cell shape uniformity, marginal adhesion, single epithelial cell size, bare nuclei, bland chromatin, normal nucleoli, and mitoses. There are two categories in the data: malignant and benign. The two classes are known to be linearly separable.

(3) Wine: This is the Wine recognition data consisting of 178 instances having 13 features resulting from a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines.

(4) Diabetes: This is the diabetes data set consisting of 768 instances having 8 attributes.

5.3. Evaluation of Clustering Quality. To compare the performance of all three algorithms (SBKM, MOCK, and MOGLC) adjusted Rand index technique [38] is used. Let n_{lk} be the number of objects that are in both class u_l and cluster v_k. Let n_l and n_k be the number of objects in class u_l and cluster v_k, respectively. Under the generalized hyper geometric model, it can be shown that

$$E\left[\sum_{l,k}\binom{n_{lk}}{2}\right] = \frac{\left[\sum_l \binom{n_{l.}}{2} \cdot \sum_k \binom{n_{k.}}{2}\right]}{\binom{n}{2}}. \tag{38}$$

The adjusted Rand index [38] can be simplified to

$$\frac{\sum_{l,k}\binom{n_{lk}}{2} - \left[\sum_{l}\binom{n_{l.}}{2}\cdot\sum_{k}\binom{n_{k.}}{2}\right]/\binom{n}{2}}{(1/2)\left[\sum_{l}\binom{n_{l.}}{2}+\sum_{k}\binom{n_{k.}}{2}\right] - \left[\sum_{l}\binom{n_{l.}}{2}\cdot\sum_{k}\binom{n_{k.}}{2}\right]/\binom{n}{2}}. \quad (39)$$

Adjusted Rand index is limited to the interval [0, 1] with a value of 1 with a perfect clustering. The high value of adjusted Rand index indicates the good quality of clustering result. The average and standard deviation of adjusted Rand index for data sets produced by 20 consecutive runs of SBKM, MOCK, and MOLGC are depicted in Tables 2(a) and 2(b), respectively.

5.4. Results on Artificial Data Sets

(1) Data set-1: We use this data set to illustrate that the proposed algorithm incorporated with line symmetry distance can also be applied to detect ring-shaped clusters even if they are crossed. Figure 10(a) shows the clustering result achieved by the SBKM algorithm. Figure 10(b) illustrates the final result achieved by the MOCK algorithm. Figure 10(c) shows the clustering result of the MOLGC algorithm. We find that the SBKM algorithm cannot work well for this case. Both MOLGC and MOCK clustering algorithms provide $K = 2$ as the optimal number of clusters in different runs. SBKM clustering algorithm discovers $K = 2$ number of clusters but it is unable to perform the proper partitioning from this data set in different runs.

(2) Data set-2: This data set is a combination of ring-shaped, compact, and linear clusters, as shown in Figure 9(b). Most clustering algorithms based on objective function minimization fail to detect this kind of data sets because their performance depends on the dissimilarity measures used to generate a partition of the data sets. The clustering result achieved by the SBKM algorithm is shown in Figure 11(a). The final clustering result of the MOCK algorithm is illustrated in Figure 11(b). Figure 11(c) shows that the proposed algorithm works well for a set of clusters of different geometrical structures. Both SBKM and MOCK clustering algorithms provide $K = 3$ number of clusters in different runs but both are unable to perform the proper partitioning from this data set. MOLGC clustering algorithm detects $K = 3$ the optimal number of clusters and the proper partitioning from data set-2 in all consecutive runs.

(3) Data set-3: As can be seen from Figure 12(a) to Figure 12(c), for this data set the SBKM clustering technique is unable to detect appropriate number of clusters. The best solution provided by MOCK is not able to determine the appropriate number of clusters from this data set. The corresponding partitioning is shown in Figure 12(b). MOLGC algorithm is able to detect $K = 5$ the appropriate number of clusters from this data set in different runs. The corresponding partitioning is shown in Figure 12(c). MOCK splits data points of one cluster into two clusters and provides $K = 6$ as the optimal number of clusters in different runs. SBKM merges the all data points into four clusters and provides $K = 4$ as the appropriate number of clusters.

(4) Data set-4: Both MOCK and MOLGC clustering algorithms are able to detect $K = 4$ the appropriate number of clusters from this data set in different runs. The clustering result obtained by the SBKM algorithm is shown in Figure 13(a). The partitioning identified by MOCK clustering algorithm is shown in Figure 13(b). Figure 13(c) shows that the proposed algorithm works well for this data set. SBKM again overlaps the data points in two clusters and discovers $K = 4$ as the optimal number of clusters. It is unable to perform the proper partitioning from this data set in different runs.

(5) Data set-5: The proposed MOLGC algorithm and MOCK algorithms are able to detect $K = 4$ the appropriate number of clusters from this data set in different runs. MOCK merges the data points of two clusters and it is not able to detect proper partitioning from this data set in all runs. SBKM is not able to detect $K = 5$ the appropriate number of clusters and the proper partitioning from this data set in different runs. It again splits data points of one cluster into two clusters and provides $K = 5$ clusters. As shown in the Tables 2(a) and 2(b), the SBKM algorithm cannot work well.

(6) Data set-6: From Tables 2(a) and 2(b), it is clear that proposed MOLGC and MOCK algorithms perform much better on this data set than the other algorithm SBKM. SBKM detects $K = 12$ clusters from this data set. It is unable to provide the appropriate number of clusters and the proper partitioning in different runs. Both MOCK and MOLGC clustering algorithms detect $K = 10$ the appropriate number of clusters from data set-6 in all runs. But MOCK performs overlapping on some data points into two clusters from this data set.

(7) Data set-7: As can be seen from Table 2(a), it is noticeable that MOLGC performs the best (providing the highest adjusted Rand index value) for this data set. The performance of MOCK is also better when compared to SBKM algorithm. For this data set, SBKM provides $K = 6$ as the optimal number of clusters. It splits the maximum dense clusters into two clusters and overestimates the number of clusters from this data set. Both MOCK and MOLGC clustering algorithms produce $K = 4$ the proper number of clusters and partitioning from this data set in different runs. But adjusted Rand index value corresponding to the partitioning obtained by MOLGC is higher than that of MOCK (as shown in Table 2(a)).

(8) Data set-8: As shown in Table 2(a), the adjusted Rand index of MOLGC is the highest for this data set, while

Table 2: (a) Average of adjusted Rand index for SBKM, MOCK, and MOLGC. (b) Standard deviation of adjusted Rand index for SBKM, MOCK, and MOLGC.

(a)

Data sets	Number of points (N)	Number of dimensions (D)	Number of clusters (K)	Average value of adjusted Rand index		
				SBKM	MOCK	MOLGC
Data set-1	200	2	2	0.7585	0.9840	0.9845
Data set-2	350	2	3	0.7491	0.9245	0.9515
Data set-3	250	2	5	0.5158	0.9510	0.9910
Data set-4	1000	2	4	0.8605	0.9815	0.9816
Data set-5	838	10	4	0.7225	0.9862	0.9895
Data set-6	3050	10	10	0.6585	0.9673	0.9795
Data set-7	351	50	4	0.6775	1.0000	1.0000
Data set-8	2328	50	10	0.6325	0.9950	0.9955
Iris	150	4	3	0.7685	0.9350	0.9810
Cancer	683	9	2	0.7877	0.9520	0.9740
Wine	178	13	3	0.6591	0.9575	0.9585
Diabetes	768	8	2	0.7111	0.9840	0.9910

(b)

Data sets	Number of points (N)	Number of dimensions (D)	Number of clusters (K)	Standard deviation of adjusted Rand index		
				SBKM	MOCK	MOLGC
Data set-1	200	2	2	0.075	0.040	0.035
Data set-2	350	2	3	0.081	0.055	0.045
Data set-3	250	2	5	0.090	0.078	0.050
Data set-4	1000	2	4	0.121	0.095	0.055
Data set-5	838	10	4	0.125	0.085	0.060
Data set-6	3050	10	10	0.150	0.070	0.028
Data set-7	351	50	4	0.175	0.075	0.041
Data set-8	2328	50	10	0.190	0.090	0.035
Iris	150	4	3	0.080	0.036	0.022
Cancer	683	9	2	0.090	0.045	0.037
Wine	178	13	3	0.085	0.051	0.035
Diabetes	768	8	2	0.070	0.050	0.030

the performance of MOCK is second. However, the performance of SBKM algorithm is found poor. For this data set, both SBKM and MOCK detect $K = 11$ as the appropriate number of clusters but both clustering algorithms are unable to produce the appropriate partitioning from this data set in all consecutive runs. The adjusted Rand index values reported in Tables 2(a) and 2(b) also show the poorer performance of both SBKM and MOCK algorithms from this data set. MOLGC discovers $K = 10$ as appropriate number of clusters and the appropriate partitioning from this data set in different runs.

5.5. Results on Real Data Sets

(1) Iris: As seen from Table 2(a), the adjusted Rand index of MOLGC is the best for Iris, while the performance of MOCK is second. However, it can be seen from Tables 2(a) and 2(b) that the performance of SBKM algorithm is found poor. SBKM, MOCK, and MOLGC provide $K = 3$ as the appropriate number of clusters form this data set in all consecutive runs. But SBKM detects overlapping of data points in two clusters whereas the third cluster is well separated from these two clusters.

(2) Cancer: As can be seen from Table 2(a), it is manifest that MOLGC performs the best (providing the highest adjusted Rand index value) for this data set. The performance of MOCK and MOLGC is similar, but the performance of SBKM algorithm is found poor. All clustering algorithms are able to provide $K = 2$ the proper number of clusters from this data set in different consecutive runs.

(3) Wine: From Tables 2(a) and 2(b), it is evident that MOLGC performs the best for this data set. Both

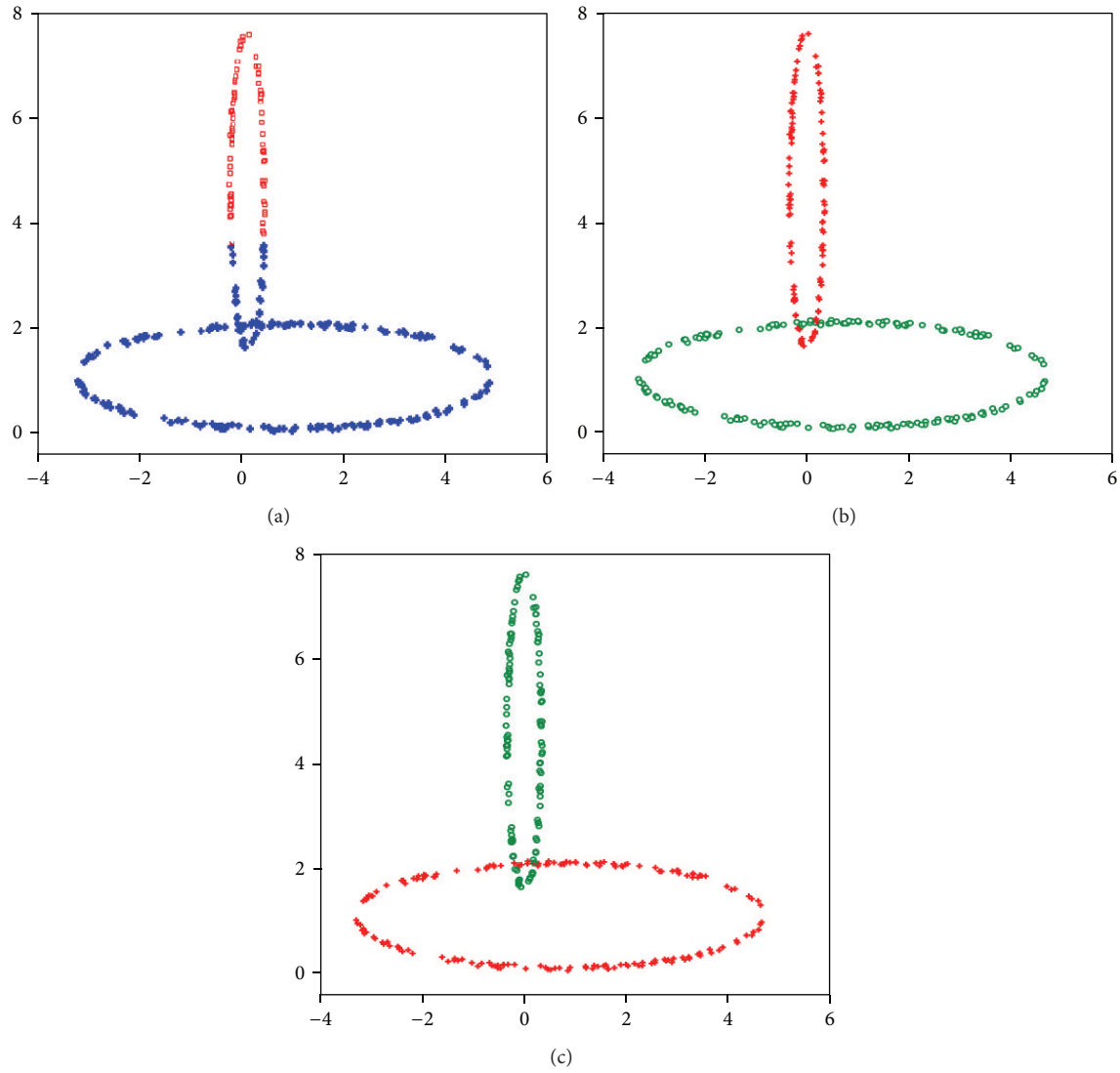


Figure 10: (a) The clustering result achieved by the SBKM (data set-1). (b) The clustering result achieved by the MOCK (data set-1). (c) The clustering result achieved by the MOLGC (data set-1).

MOLGC and MOCK clustering algorithms are able to provide $K = 3$ as the proper number of clusters from this data set. The adjusted Rand index value obtained by MOLGC is also the maximum (refer Table 2(a)). SBKM is not able to perform the proper partitioning from this data set.

(4) Diabetes: From Tables 2(a) and 2(b), it is again clear that MOLGC performs much better than the other two algorithms (providing the highest adjusted Rand index value). MOLGC and MOCK clustering algorithms detect $K = 2$ as the optimal number of clusters from this data set. Both clustering algorithms are able to provide the proper partitioning from this data set in different consecutive runs. SBKM is not able to detect appropriate number of clusters in all consecutive runs. The corresponding adjusted Rand index value is reported in Tables 2(a) and 2(b).

It can be seen from the above results that the proposed MOLGC clustering algorithm is able to detect the appropriate number of clusters from most of the data sets used here for the experiments. The superiority of MOLGC is also established on four real-life data sets which are of different characteristics with the number of dimensions varying from 2 to 13. Results on the eight artificial and four real-life data sets establish the fact that MOLGC is well-suited to detect the number of clusters from data sets having clusters of widely varying characteristics.

The performance results reported in Tables 2(a) and 2(b) clearly demonstrate the clustering accuracy of SBKM, MOCK, and MOLGC for artificial and real data sets. Table 3 indicates average computing time taken by 20 consecutive runs of SBKM, MOCK, and MOLGC for clustering of the above data sets. Results show SBKM and MOCK execution time is increased linearly with increasing dimensions of data

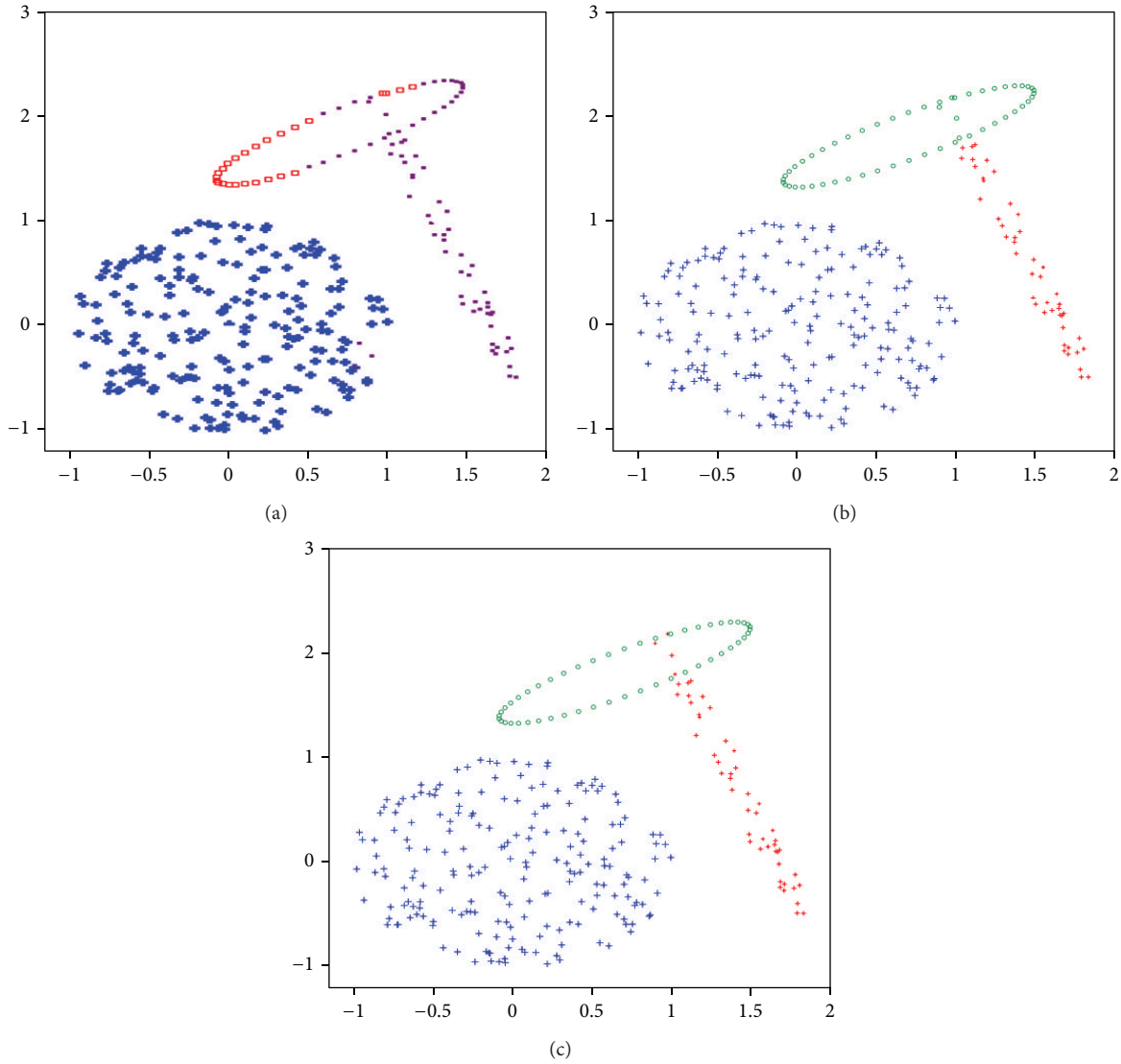


Figure 11: (a) The clustering result achieved by the SBKM (data set-2). (b) The clustering result achieved by the MOCK (data set-2). (c) The clustering result achieved by the MOLGC (data set-2).

Table 3: Comparison of the execution time (in seconds).

Data sets	SBKM	MOCK	MOLGC
Data set-1	10	15	18
Data set-2	12	18	20
Data set-3	25	35	40
Data set-4	40	45	48
Data set-5	70	52	50
Data set-6	1450	378	365
Data set-7	246	156	150
Data set-8	5300	656	640
Iris	10	14	14
Cancer	62	45	42
Wine	55	30	28
Diabetes	60	35	32

sets. The MOLGC shows better results in terms of reduction in CPU time in comparison to SBKM and MOCK.

The proposed MOLGC clustering algorithm is able to identify automatically the appropriate number of clusters in different runs. MOLGC generates the entire set of solutions with automatic determination of correct number of clusters in a single run. It consistently generates the proper cluster number from eight artificial and four real data sets in different runs.

5.6. Reconstruction Criterion. In this paper a reconstruction criterion is used to optimize the performance of the SBKM, MOCK, and MOLGC clustering algorithms. A fuzzy C-means (FCM) algorithm based clustering platform is considered in [39]. The objective of this work is to raise awareness about the essence of the encoding and decoding processes

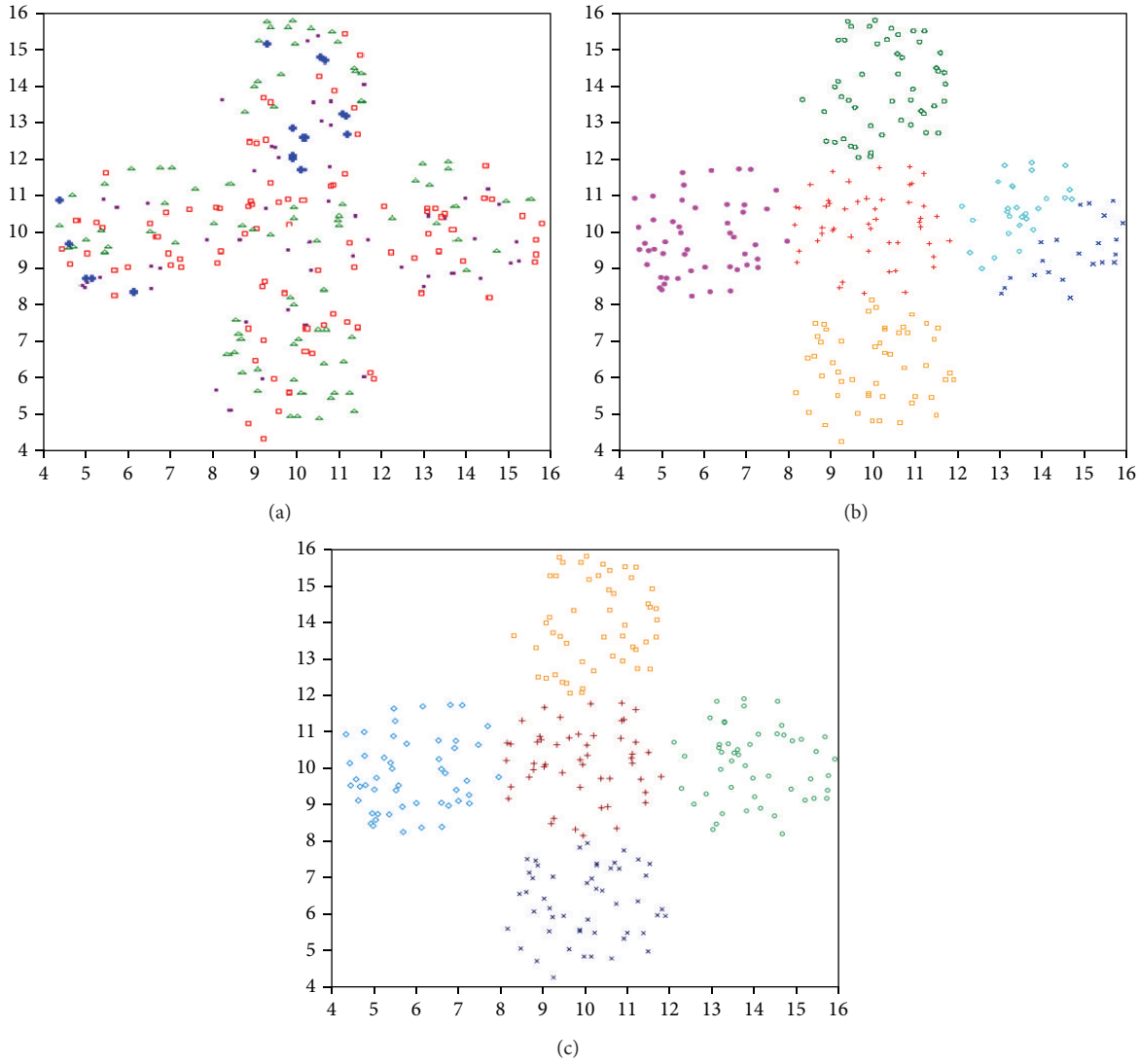


Figure 12: (a) The clustering result achieved by the SBKM (data set-3). (b) The clustering result achieved by the MOCK (data set-3). (c) The clustering result achieved by the MOLGC (data set-3).

completed in the context of fuzzy sets. The main design aspects deal with the relationships between the number of clusters and the reconstruction properties and the resulting reconstruction error. Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of n points in a multidimensional experimental data set. Now three sets of prototypes $(v_1, v_2, \ldots, v_c)$ are generated by running the SBKM, MOCK, and MOLGC clustering algorithms separately on experimental data. For any data point x_i, we obtain its membership grades to the corresponding clusters. They are denoted by u_{ij} ($i = 1, 2, \ldots, c$ and $j = 1, 2, \ldots, n$) and are result of the minimization of the following objective function:

$$\sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^{m} d^2\left(v_i, v_j\right), \tag{40}$$

where m ($m > 1$) is a coefficient. The distance d used in the objective function is viewed as the Point Symmetry Distance (PSD) in SBKM [7], nearest neighbor consistency (MOLGC), and line symmetry distance in MOLGC:

$$\sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij} = 1. \tag{41}$$

By solving (41) through the use of Lagrange multipliers, we arrive at the expression of the granular representation of the numeric value:

$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(d\left(v_i, x_j\right)/d\left(v_k, x_j\right)\right)^{2/(m-1)}}. \tag{42}$$

Figure 14 highlights the essence of reconstruction criterion. Our starting point is the result of clustering expressed in terms of the prototypes and the partition matrix.

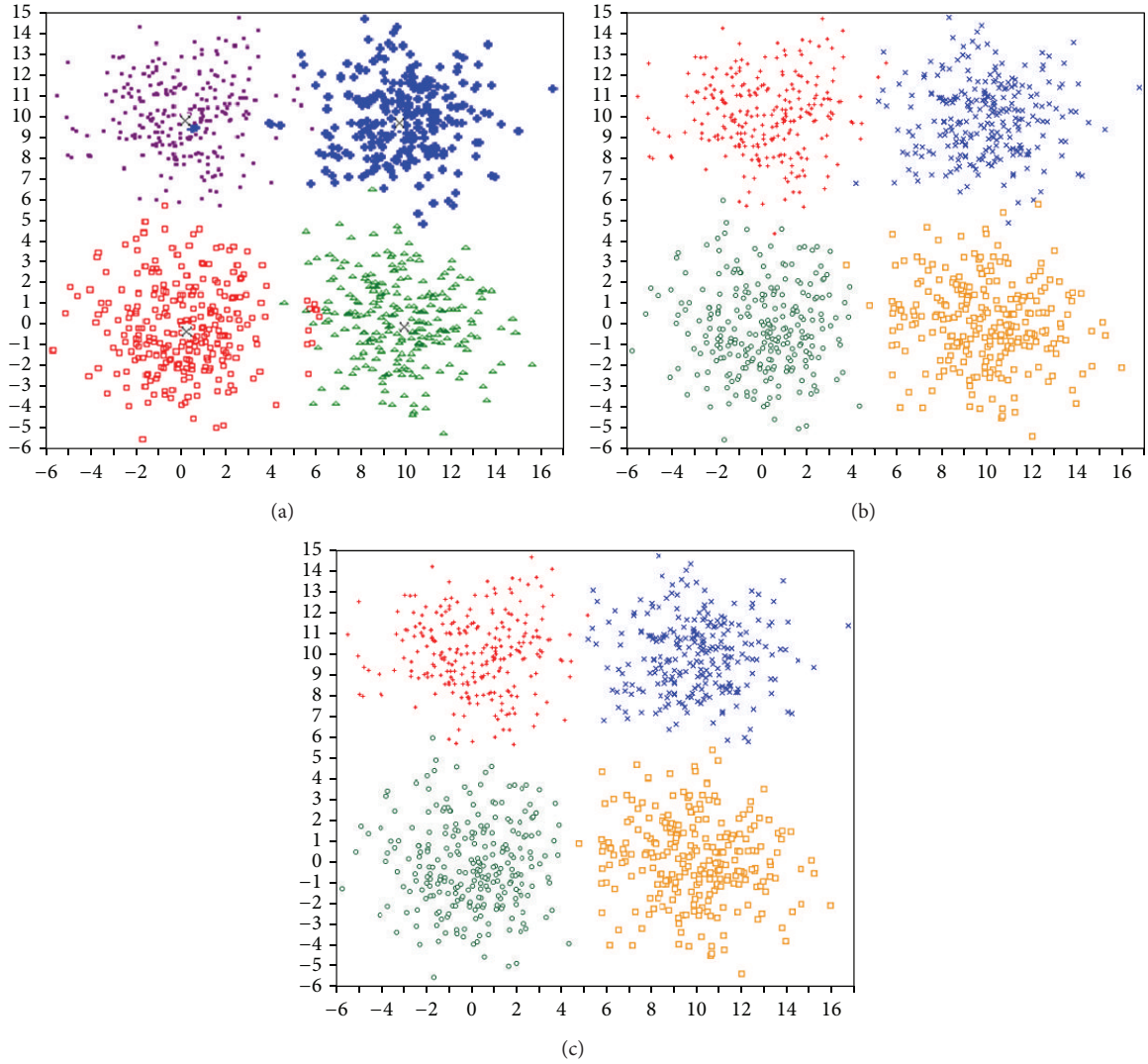


Figure 13: (a) The clustering result achieved by the SBKM (data set-4). (b) The clustering result achieved by the MOCK (data set-4). (c) The clustering result achieved by the MOLGC (data set-4).

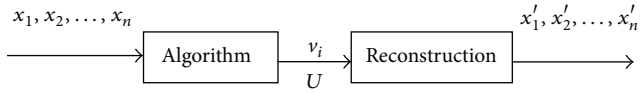


Figure 14: Scheme of the reconstruction criterion.

The main objective of this reconstruction process is to reconstruct the original data using the cluster prototypes and the partition matrix by minimizing the sum of distances [39]:

$$F = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^{m} d^2 \left(v_i, x'_j\right), \tag{43}$$

where x'_j is the reconstructed version of x_j. We used the Point Symmetry Distance (PSD) for SBKM [7], nearest neighbor consistency for MOCK, and line symmetry distance for MOLGC in (43) and zeroing the gradient of F with respect to x_k, we have

$$x'_j = \frac{\sum_{i=1}^{c} u_{ij}^{m} v_i}{\sum_{i=1}^{c} u_{ij}^{m}}. \tag{44}$$

The performance of reconstruction is expressed as

$$E = \sum_{J=1}^{n} \left\| x_j - x'_j \right\|^2. \tag{45}$$

We investigate the behavior of the clustering results quantified in terms of the criteria of reconstruction for artificial and real data sets. Table 4 presents the reconstruction error values reported for clusters by 20 consecutive runs of SBKM, MOCK, and MOLGC, respectively. In all experiments, the value of the coefficient m was set to 2.

Table 4: Reconstruction error for the artificial and real data sets.

Data set	Cluster value	SBKM	MOCK	MOLGC
Data set-1	2	19.50	14.35	11.45
Data set-2	3	17.15	13.72	10.25
Data set-3	5	15.45	10.45	9.14
Data set-4	4	16.15	11.25	9.45
Data set-5	4	17.12	12.25	10.50
Data set-6	10	11.15	8.50	5.25
Data set-7	4	17.10	12.35	8.06
Data set-8	10	12.50	8.55	3.85
Iris	3	15.82	12.47	9.25
Cancer	2	18.45	13.25	10.15
Wine	3	17.35	13.53	8.97
Diabetes	2	19.10	11.72	9.55

5.7. Statistical Significance Test. For a more careful comparison among SBKM, MOCK, and MOLGC, a statistical significance test called Wilcoxon rank sum test [40] for independent samples has been conducted at the 5% significance level. It is a nonparametric alternative to the paired t-test. It assumes commensurability of differences, but only qualitatively: greater differences still count more, which is probably desired, but the absolute magnitudes are ignored. From the statistical point of view, the test is safer since it does not assume normal distributions. Also, the outliers have less effect on the Wilcoxon test than on the t-test. The Wilcoxon test assumes continuous differences d_i; therefore they should not be rounded to, say, one or two decimals since this would decrease the power of the test due to a high number of ties. When the assumptions of the paired t-test are met, the Wilcoxon rank test is less powerful than the paired t-test. On the other hand, when the assumptions are violated, the Wilcoxon test can be even more powerful than the t-test. Three groups corresponding to three algorithms SBKM, MOCK, and MOLGC have been created for each data set. Each group consists of the performance scores (adjusted Rand index for the artificial data and real life data) produced by 20 consecutive runs of corresponding algorithm. The median values of each group for all the data sets are shown in Table 5. The results obtained with this statistical test are shown in Table 6. To establish that this goodness is statistically significant, Table 6 reports the P values produced by Wilcoxon's rank sum test for comparison of groups (SBKM, MOCK, and MOLGC) at a time. As a null hypothesis, it is assumed that there are no significant differences between the median values of two groups. However, the alternative hypothesis is that there is significant difference in the median values of the two groups. All the P values reported in the table are less than 0.05 (5% significance level).

The smaller the P value, the stronger the evidence against the null hypothesis provided by the data. The signed rank test among algorithms MOLGC, SBKM, and MOCK for artificial data and real life data provides a P value, which is very small. This is strong evidence against the null hypothesis, indicating that the better median values of the performance metrics produced by MOLGC are statistically significant and have not occurred by chance. Similar results are obtained for all other data sets and for all other algorithms compared to MOLGC, establishing the significant superiority of the MOLGC algorithm.

Table 5: Median values of adjusted Rand index for artificial and real data sets.

Data sets	SBKM	MOCK	MOLGC
Data set-1	0.7599	0.9880	0.9895
Data set-2	0.7510	0.9297	0.9555
Data set-3	0.5199	0.9560	0.9940
Data set-4	0.8685	0.9875	0.9850
Data set-5	0.7280	0.9885	0.9915
Data set-6	0.6625	0.9690	0.9825
Data set-7	0.6805	1.0000	1.0000
Data set-8	0.6375	0.9915	0.9980
Iris	0.7715	0.9380	0.9875
Cancer	0.7901	0.9565	0.9780
Wine	0.6610	0.9605	0.9615
Diabetes	0.7157	0.9870	0.9925

Table 6: P values produced by Wilcoxon Rank test for comparing MOLGC with SBKM and MOCK.

Data sets	P values	
	SBKM	MOCK
Data set-1	$1.53E-4$	$1.65E-4$
Data set-2	$1.48E-4$	$1.70E-4$
Data set-3	$1.70E-3$	$1.21E-4$
Data set-4	$1.10E-4$	$5.14E-5$
Data set-5	$2.28E-3$	$1.55E-4$
Data set-6	$2.45E-3$	$6.55E-5$
Data set-7	$1.19E-4$	$1.85E-5$
Data set-8	$1.41E-4$	$2.25E-5$
Iris	$2.31E-4$	$1.32E-4$
Cancer	$1.35E-5$	$1.65E-5$
Wine	$1.20E-4$	$5.75E-5$
Diabetes	$1.30E-4$	$1.25E-4$

6. Conclusion

In this paper, a line symmetry based multiobjective MOLGC algorithm is proposed. In the proposed algorithm, the points are assigned to different clusters based on line symmetry based distance. In this multiobjective genetic clustering algorithm two objective functions, one based on a new line symmetry based distance and another based on Euclidean distance DB index, are used for computation of fitness. The proposed algorithm can be used to group given data set into a set of clusters of different geometrical structures. Compared with the SBKM and the MOCK, the proposed MOLGC algorithm adopts a line symmetry approach to cluster data; therefore, the later approach is more flexible. Most importantly, a modified version of the line symmetry distance is proposed to overcome some limitations of

the original version of the symmetry distance introduced by Chung and Lin [8]. In addition, the MOLGC algorithm outperforms the SBKM algorithm and the MOCK based on the comparisons of the results presented in this paper. Tables 2(a) and 2(b) indicate the quality of best clustering results in terms of adjusted Rand index generated by SBKM, MOCK, and MOLGC for eight artificial data sets and four real data sets. Table 3 tabulates the comparisons of the computational time of the MOLGC algorithm and other popular clustering algorithms. Obviously, the proposed algorithm needs more computational resources than other algorithms. However, the proposed algorithm provides a possible solution to detect clusters with a combination of compact clusters, shell clusters, and line-shaped clusters. It should be emphasized that although the present MOLGC algorithm demonstrates to some extent the potential of detecting clusters with different geometrical structures, there still remains a lot of research space for improving the MOLGC algorithm, such as how to reduce the computational time.

Finally, it is an interesting future research topic to extend the results of this to face recognition.

Conflict of Interests

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

References

[1] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, San Francisco, Calif, USA, 2011.

[2] W. Pedrycz, *Knowledge-Based Clustering: From Data to Information Granules*, John Wiley & Sons, 2005.

[3] B. S. Everitt, *Cluster Analysis*, Edward Arnold, London, UK, 3rd edition, 1993.

[4] W. Miller, *Symmetry Groups and Their Applications*, Academic Press, London, UK, 1972.

[5] H. Zabrodsky, S. Peleg, and D. Avnir, "Symmetry as a continuous feature," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 12, pp. 1154–1166, 1995.

[6] K. Kanatani, "Comments on symmetry as a continuous feature," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 3, pp. 246–247, 1995.

[7] M. C. Su and C. H. Chou, "A modified version of the K-means algorithm with a distance based on cluster symmetry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 674–680, 2001.

[8] K.-L. Chung and J.-S. Lin, "Faster and more robust point symmetry-based K-means algorithm," *Pattern Recognition*, vol. 40, no. 2, pp. 410–422, 2007.

[9] J. McQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the 5th Berkeley Symposium on Mathematics, Statistics, and Probabilistic*, vol. 1, pp. 281–297, 1967.

[10] C. A. Murthy and N. Chowdhury, "In search of optimal clusters using genetic algorithms," *Pattern Recognition Letters*, vol. 17, no. 8, pp. 825–832, 1996.

[11] G. Garai and B. B. Chaudhuri, "A novel genetic algorithm for automatic clustering," *Pattern Recognition Letters*, vol. 25, no. 2, pp. 173–187, 2004.

[12] S. Vijendra, K. Ashiwini, and S. Laxman, "A fast evolutionary algorithm for automatic evolution of clusters," *Information Technology Journal*, vol. 11, no. 10, pp. 1409–1417, 2012.

[13] S. Saha and S. Bandyopadhyay, "A new point symmetry based fuzzy genetic clustering technique for automatic evolution of clusters," *Information Sciences*, vol. 179, no. 19, pp. 3230–3246, 2009.

[14] H. He and Y. Tan, "A two-stage genetic algorithm for automatic clustering," *Neurocomputing*, vol. 81, pp. 49–59, 2012.

[15] J. Handl and J. Knowles, "Evolutionary multiobjective clustering," in *Parallel Problem Solving from Nature—PPSN VIII*, vol. 3242 of *Lecture Notes in Computer Science*, pp. 1081–1091, Springer, Berlin, Germany, 2004.

[16] Y. K. Kim, K. T. Park, and J. S. Ko, "A symbiotic evolutionary algorithm for the integration of process planning and job shop scheduling," *Computers & Operations Research*, vol. 30, no. 8, pp. 1151–1171, 2003.

[17] J. Handl and J. Knowles, "An evolutionary approach to multiobjective clustering," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 1, pp. 56–76, 2007.

[18] S. Saha and S. Bandyopadhyay, "A symmetry based multiobjective clustering technique for automatic evolution of clusters," *Pattern Recognition*, vol. 43, no. 3, pp. 738–751, 2010.

[19] W. Gong, Z. Cai, C. X. Ling, and B. Huang, "A point symmetry based automatic clustering approach using differential evolution," in *Advances in Computation and Intelligence: Proceedings 4th International Symposium, ISICA 2009 Huangshi, China, Ocotober 23-25, 2009*, vol. 5821 of *Lecture Notes in Computer Science*, pp. 151–162, Springer, Berlin, Germany, 2009.

[20] K. Suresh, D. Kundu, S. Ghosh, S. Das, and A. Abraham, "Automatic clustering with multi-objective differential evolution algorithms," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 2590–2597, May 2009.

[21] K. Suresh, D. Kundu, S. Ghosh, S. Das, A. Abraham, and S. Y. Han, "Multi-objective differential evolution for automatic clustering with application to micro-array data analysis," *Sensors*, vol. 9, no. 5, pp. 3981–4004, 2009.

[22] W. Pedrycz, *Granular Computing: An Emerging Paradigm*, vol. 70 of *Studies in Fuzziness and Soft Computing*, Springer, 2001.

[23] S. Bandyopadhyay and S. Saha, "A point symmetry-based clustering technique for automatic evolution of clusters," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 11, pp. 1441–1457, 2008.

[24] S. Saha and U. Maulik, "A new line symmetry distance based automatic clustering technique: application to image segmentation," *International Journal of Imaging Systems and Technology*, vol. 21, no. 1, pp. 86–100, 2011.

[25] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Prentice Hall, Englewood Cliffs, NJ, USA, 2nd edition, 2002.

[26] J. H. Freidman, L. B. Jon, and A. F. Raphael, "An algorithm for finding best matches in logarithmic expected time," *ACM Transactions on Mathematical Software*, vol. 3, no. 3, pp. 209–226, 1977.

[27] K. Köser, V. Härtel, and R. Koch, "Robust feature representation for efficient camera registration," in *Pattern Recognition*, vol. 4174 of *Lecture Notes in Computer Science*, pp. 739–749, Springer, Berlin, Germany, 2006.

[28] H. Jégou, M. Douze, and C. Schmid, "Improving bag-of-features for large scale image search," *International Journal of Computer Vision*, vol. 87, no. 3, pp. 316–336, 2010.

[29] P. Wu, S. C. Hoi, D. D. Nguyen, and Y. W. He, "Randomly projected KD-trees with distance metric learning for image retrieval," in *Advances in Multimedia Modeling: 17th International Multimedia Modeling Conference, MMM 2011, Taipei, Taiwan, January 5–7, 2011, Proceedings, Part II*, vol. 6524 of *Lecture Notes in Computer Science*, pp. 371–382, Springer, Berlin, Germany, 2011.

[30] R. Chaudhry and Y. Ivanov, "Fast approximate nearest neighbor methods for example-based video search," in *Video Analytics for Business Intelligence*, vol. 409 of *Studies in Computational Intelligence*, pp. 69–97, Springer, Berlin, Germany, 2012.

[31] C. S. Anan and R. Hartley, "Optimised KD-trees for fast image descriptor matching," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '08)*, pp. 1–8, June 2008.

[32] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *Proceedings of the 4th International Conference on Computer Vision Theory and Applications (VISAPP '09)*, vol. 1, pp. 331–340, February 2009.

[33] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, no. 2, pp. 224–227, 1979.

[34] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Mich, USA, 1975.

[35] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, no. 4, pp. 656–667, 1994.

[36] J. J. Grefenstette, "Optimization of control parameters for genetic algorithms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 16, no. 1, pp. 122–128, 1986.

[37] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, 1989.

[38] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, no. 1, pp. 193–218, 1985.

[39] W. Pedrycz and J. V. de Oliveira, "A development of fuzzy encoding and decoding through fuzzy clustering," *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no. 4, pp. 829–837, 2008.

[40] M. Hollander and D. A. Wolfe, *Nonparametric Statistical Methods*, John Wiley & Sons, New York, NY, USA, 2nd edition, 1999.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 967320, 11 pages
http://dx.doi.org/10.1155/2015/967320

Research Article

Optimization of High-Dimensional Functions through Hypercube Evaluation

Rahib H. Abiyev[1] and Mustafa Tunay[2]

[1]*Applied Artificial Intelligence Research Centre, Near East University, P.O. Box 670, Lefkosa, Northern Cyprus, Mersin 10, Turkey*
[2]*Computer and Instructional Technologies Education, Eastern Mediterranean University, Famagusta, Northern Cyprus, Mersin 10, Turkey*

Correspondence should be addressed to Rahib H. Abiyev; rahib.abiyev@neu.edu.tr

Received 26 September 2014; Revised 7 February 2015; Accepted 10 February 2015

Academic Editor: Piotr Franaszczuk

A novel learning algorithm for solving global numerical optimization problems is proposed. The proposed learning algorithm is intense stochastic search method which is based on evaluation and optimization of a hypercube and is called the hypercube optimization (HO) algorithm. The HO algorithm comprises the initialization and evaluation process, displacement-shrink process, and searching space process. The initialization and evaluation process initializes initial solution and evaluates the solutions in given hypercube. The displacement-shrink process determines displacement and evaluates objective functions using new points, and the search area process determines next hypercube using certain rules and evaluates the new solutions. The algorithms for these processes have been designed and presented in the paper. The designed HO algorithm is tested on specific benchmark functions. The simulations of HO algorithm have been performed for optimization of functions of 1000-, 5000-, or even 10000 dimensions. The comparative simulation results with other approaches demonstrate that the proposed algorithm is a potential candidate for optimization of both low and high dimensional functions.

1. Introduction

One of the basic problems of numerical optimization techniques is the computing globally optimal solutions of high-dimensional functions. The aim of optimization is the finding of optimum values of the objective function through learning the parameters of the function given in the defined domains. The learning algorithms are basically divided into two categories. The algorithms based on derivatives of the cost functions (or objective functions) are called derivative based learning algorithms, and the algorithms that do not use the derivatives of the cost functions are called derivative free learning. Recently various learning techniques have been applied to obtain the solution of different optimization problems. However, derivative based learning techniques do not fare well for finding global optimal solutions of the nonlinear problems having many local optimal solutions. Derivative free learning techniques and evolutionary computing are effective optimization techniques that can be used to solve "local minima" problem and find global optimum of the problem.

In the literatures, various learning algorithms have been applied to find global optimal solution. Monte-Carlo method [1], Vegas algorithm [2], and Cat algorithm [3] are extensively used for solution of different optimization problems. Some of more used algorithms are genetic algorithms (GA) [4, 5], evolution strategies [6], differential evolution (DE) [7], particle swarm optimization [8], and other nonevolutionary methods such as simulated annealing [9], tabu search [10], ant-colony optimization (ACO) [11], and artificial bee colony algorithm [12]. The integration of the methods with computational intelligence techniques is widely used to solve different practical problems of engineering and science [13–18].

Recently number of researches has been done on global optimization, but there are still not many powerful techniques for optimization of dense high-dimensional problems. This is because the global optimization of high-dimensional functions is computationally expensive, cost involved. These

problems are characterized by many parameters, and many iterations and arithmetic operations are needed for evaluations of these functions. In practical applications, evaluation of the function is often very expensive and large number of function evaluations might not be very feasible [19].

Some learning algorithms have been designed for global optimization of high-dimensional functions. Reference [20] uses new variant of differential evolution (DE), named DECC-I and DECC-II for high-dimensional optimization (up to 1000 dimensions). The algorithms use several novel strategies that focus on problem decomposition and subcomponents cooperation. An improved differential evolution algorithm [21], self-adaptive differential evaluation algorithm [22], differential ant-stigmergy [23], particle swarm optimization [24, 25], modified multiscale particle swarm optimization [26], surrogate-assisted evolutionary programming [27], and group search optimizer (GSO) inspired by animal behavior [28] are designed and applied for global optimization of high-dimensional functions. As shown the designed algorithms are basically modification, improvement, and adaptation of existing evolutionary algorithms in particularly DE, PSO, and GA. Using these methods the researchers try to obtain reasonable results for optimization functions. In spite of some success, these techniques are still not very much suitable for high-dimensional global optimization problems [19]. The proposed algorithms are more suitable for low-dimensional problems. The dimension that was used in above research papers was maximum 100 and some of them 1000. In this paper, the novel method that solves high-dimensional global optimization problems having sizes of 1000, 5000, and 10000 is proposed. The proposed novel method is called hypercube optimization (HO) algorithm. The HO algorithm is based on designing hypercube, selecting the best elements and applied them to multivariate systems for optimization of the objective function. This algorithm approaches optimal points using the best elements determined during learning.

The paper is organized as follows. Section 2 presents the hypercube optimization algorithm proposed. The processes used in the algorithm are described. Section 3 describes the test functions used in simulations. Section 4 includes application of the algorithm on test functions. Section 5 presents comparative results of HO algarithm with some existing methods. Finally, in Section 6 conclusions are presented.

2. Hypercube Optimization Algorithm

The HO algorithm is an evolutionary algorithm that takes inspiration from the behaviour of a dove discovering new areas for food in natural life. In such behaviour a flying dove searches for new locations of food. The dove flies down in a unique way and marks the area that may have food. The dove flies up again and it chooses the previously marked areas and changes and shrinks the sizes of the search area. In a search process, the dove is not limited to a single area. The dove picks new search area according to the density of food (domain for the objective function). The dove stops flying and keeps in mind the area which has food. After eating the food, the dove is looking for a new search area. The dove jumps or flies down another area branch to find a new area. The dove does not fly to another area when it gets to an area that has the most food.

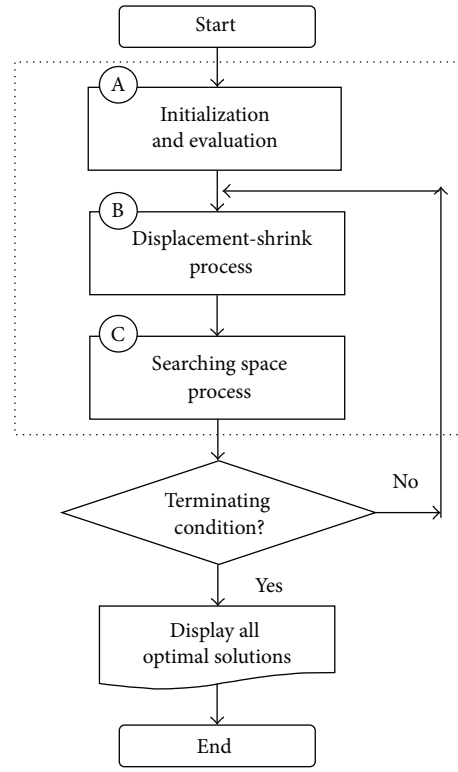


Figure 1: Flowchart of the hypercube optimization algorithm. Here A is *initialization and evaluation*, B is *displacement-shrink*, and C is *searching space* processes.

In the paper, the hypercube is used to describe the search area. Inside the search area, the value of an objective function is evaluated according to the quantity and density of food. Next, the functional distances between each of two solutions are determined. This distance helps the algorithm to determine the next new search area. This is performed using the displacement-shrink process.

The hypercube optimization algorithm is a derivative-free learning method based on evaluation of set of points randomly distributed in an m-dimensional hypercube. After evaluation the point shifts and contracts according to the average between previous best points in order to determine new best points inside the hypercube. The contraction is greater when the movement is smaller to accelerate the convergence. This operation will be reported as an optimal solution at the end of the iterations.

The HO algorithm is an intense stochastic search method based on hypercube (HC) evaluation. The general structure regarding the visualization of the flowchart of the hypercube optimization algorithm is illustrated in Figure 1. As shown from the figure, the HO algorithm includes three basic processes.

Step A (initialization and evaluation process). The algorithm begins with the generation of a hypercube and initialization of matrices and variables within the hypercube. Here

the hypercube is represented by the center and size (radii). The new points with uniform distribution are randomly generated within the hypercube. It proceeds to the through main loop, by which convergence to the global minimum is sought, and it finishes when any of the termination criteria is fulfilled.

Step B (displacement-shrink process). The displacement-shrink process is deployed to find the new best point. This is implemented by computing the average of the current best point and the previous best one. The average between both values is taken as a conservative measure to avoid excessive fluctuations in the search.

Step C (searching space process). The searching space process controls the movements of X solutions according to the defined interval (commonly $[0, 0.1]$). The searching space process initializes a new hypercube and repeats the whole process.

The initialization and evaluation process, displacement-shrink process, and searching space process are repeated in each learning iteration. While specific termination conditions are satisfied the whole processes are continued to execute.

At each iteration, the newly generated hypercube changes and shrinks its sizes until the optimum points are located. Unlike other methods, like particle swarm optimization, the points in the hypercube optimization algorithm do not move according to a specific rule nor does the method record them, except for the best points. This permits a rapid selection of a new best zone and an intense search in it. Thus, the hypercube optimization algorithm does not perform any local search but rather it is always global. This behavior allows the algorithm to move rapidly to globally best points, as it does not waste time in local searches.

Following in the next subsections the descriptions of each step are presented in detail.

2.1. Hypotheses and Representation of Solution. As in all real-valued single-objective unconstrained optimization algorithms, we try to find the minimum (or equivalently the maximum) of a scalar objective function $f_{(x)}$ and represent the free parameters as a vector or point $X = (x_1, x_2, x_3, \ldots, x_m)$, where m is the dimension of the problem. Therefore, f is a mapping $\mathfrak{R}^m \rightarrow \mathfrak{R}$. We assume the following hypotheses.

(i) f is available only as a black box; that is, we have no knowledge or possibility of control of its interior functions. We access f only via input-output.

(ii) f has a continuous domain inside the bounds; that is, every point inside the bounds has a mapping by f.

(iii) f is well-behaved in the domain, at least numerically; that is, it is continuous and presents certain smoothness. This constrains overly noisy functions, where there is no spatial correlation. But implicit is also the assumption of some noisiness, whereby finite differences in the neighborhood of a point are not similar to the derivatives of the noiseless function.

Table 1: Initial points.

Symbol	Definition
m	Dimension of hypercube
R	radii of HC
X_c	Center of 1st HC (zone)
$X = X_0$	Take initial point as 1st HC
LB, UB	Lower and upper bounds of first HC (zone)
N	Number of points in each HC
X	$N \times m$ points, solutions
F	$N \times 1$ points, values of functions
F_{best}	Best value of objective function
Create matrices: $X(N \times m)$ $F(N \times 1)$ F_{best}: best value of objective function	

(iv) The number of searching points (N) is enough for correctly sampling f's domain (related to the previous point). Therefore, N is directly related to the dimension of the problem (m) and f's smoothness.

2.2. Initialization and Evaluation Process. Initialization and evaluation is the first block of hypercube optimization algorithm. The starting conditions are

(1) initial (and global) boundaries for all points: these boundaries are the sides of the hypercube;

(2) initialization of solutions inside the hypercube and an initial random choice of a best point X_0 (if not available, the central point of the initial hypercube will be taken) in the given set.

Initial points of the hypercube optimization algorithm are presented in Table 1. At the starting stage the data radii and centre of the HC are generated randomly and these parameters are used to initialize the first HC. Then uniformly distributed N searching points are generated inside the hypercube. Using these points, the values of the objective function are determined. Here the concept is to have an approximate knowledge about the location of the lowest values of f. This initial sampling has to be sufficiently dense so as to probe all the possible zones of higher and lower values; otherwise, the algorithm can take the zone sought (global optimum) as a simply better one (local optimum). As pointed out above, this density (and hence the number of points N) is a function of the dimension m and the smoothness of the function. The problems with higher dimension will require higher N.

The hypercube optimization algorithm begins with the initialization of matrices and variables; it proceeds to the main loop, by which convergence to the global minimum is sought, and it finishes when any of the termination criteria is fulfilled. The details regarding the visualization of the flowchart initialization and evaluation of the HO algorithm

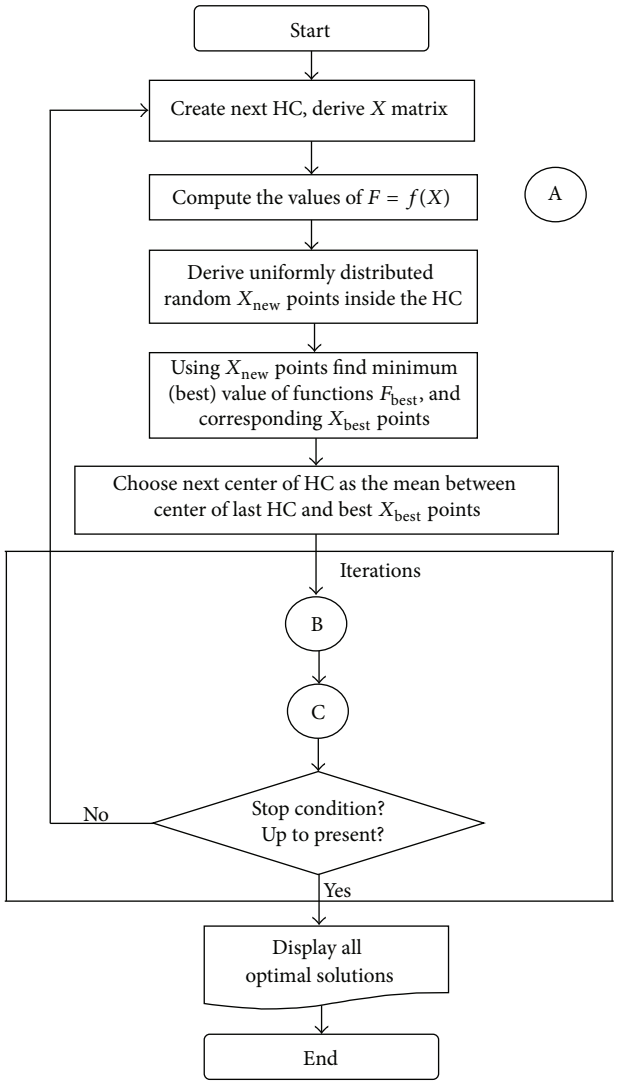


Figure 2: Flowchart of the initialization and evaluation process. Here B is displacement-shrink process and C is searching space process.

are illustrated in Figure 2. After the start block, initial point X_0 is generated as the centre of the first hypercube (HC). The initial value of the radii of the first HC is determined according to the change interval of the test (objective) functions. Next using the value of centre X_0 the dimension of the hypercube is derived according to formula (1). After creating the hypercube, the X matrix is generated within this hypercube. The size of X is defined by $(N \times m)$. N is a number of generated points. We need to comment that in future iterations ($i = 2, 3, \ldots$) the hypercube is created using the values of X matrix.

We have illustrated this process as follows with initial points to create them with default values.

(1) Dimension of hypercube is

$$m = \text{length}\,(X_0)\,. \tag{1}$$

(2) Row vectors with lower and upper boundaries of HC are

$$\begin{aligned} \text{LB} &= \min\,(X \text{ bounds})\,, \\ \text{UB} &= \max\,(X \text{ bounds})\,. \end{aligned} \tag{2}$$

(3) Dimensions of m-dimensional HC's are

$$D = \text{UB} - \text{LB}. \tag{3}$$

(4) Central values are

$$X_c = \frac{(\text{LB} + \text{UB})}{2}. \tag{4}$$

(5) Vector with radii of HC is

$$\begin{aligned} R_0 &= \frac{D}{2}, \\ R &= R_0. \end{aligned} \tag{5}$$

According to X matrix, the row vector with lower and upper boundaries of the hypercube (2) is determined. Using these boundaries, obtained from the first hypercube (zone), the radii (4) and the centre (5) points of the next hypercube are determined. X matrix, defined as N searching points, is applied to determine the values of the test function, that is, $F(f(x))$ matrix, as pointed out above in Table 1. In the next step using the HC, the new uniformly random points are derived. The number of points is defined according to the dimension of the HC. These points form the new X_{new} matrix. This matrix is used to evaluate the test functions. As a result of evaluation, the best (minimum) value of function F_{best} and the corresponding X_{best} points are determined. By "best" we mean the vector that corresponds to the best fitness (e.g., the lowest objective function value for a minimization problem) in the entire population at ith iteration. The X_{best} point is improved (updated) using local search; that is, $X_{best}^{new} = X_{best} + \rho\Delta F$. Here $0 \le \rho \le 1$, F is the objective function. The improvement is continued until ΔF becomes acceptably small value less than a preset value (tol$F$). The derived best points are used to determine the centre and the radii of the next hypercube. This operation is realized by calculating the mean of the center of the last HC (X_{last_centre}) and the previous best (X_{best}) points; that is, $(X_{last\_centre} + X_{best})/2$. This process is called "displacement." As shown the created second HC is derived from the previous HC and the sizes of the second HC will be less than the sizes of the previous one. In future operations, the last-second HC will be used to create the next-third hypercube.

In summary, we can unify the evaluation and learning processes as follows. When the new hypercube is initialized, the function is evaluated at new points, randomly (with uniform distribution) chosen from inside of the hypercube. The new minimum is determined and compared with the last minimum. If the new minimum is worse (greater) than the previous one, then a new iteration will be started. If the same value is repeated several consecutive times then

the algorithm ends, and the best minimum is considered as the global minimum.

After the above given initialization and evaluation processes the implementation of displacement-shrink process and searching space process is performed. The whole process is repeated until specific termination conditions are satisfied.

2.3. Displacement and Shrink Process. The center of the next hypercube will be just the average between the current best point and the previous one; that is, $(X_{\text{last\_centre}} + X_{\text{best}})/2$. The average between both values is taken as a conservative measure to avoid excessive fluctuations in the search and to prevent moving suddenly to a neighboring zone where a lower value was found, but which perhaps is just a local minimum. The radii of the new hypercube are determined as $R_{\text{new}} = R_{\text{old}} * S$. Here S is a factor of convergence which is defined in the next section (see (10)).

In addition to moving, the hypercube has to contract in order to refine the search and to converge to a unique and certain—assumed global—minimum. This contraction is controlled by the movement of the average of best values. For large displacements, there is no contraction, as we interpret that the global minimum is still very uncertain. For small or null displacements, the hypercube will shrink, as we interpret this to mean that we are closer to the global minimum: the contraction is greater for smaller movements. This derives the fast convergence of the method, while it prohibits getting stuck at undesired (local) minima.

The details regarding the visualization of the flowchart of the displacement-shrink process of the hypercube optimization algorithm is illustrated in Figure 3. At first, the minimum of value of F_{best} is compared with the new value of F_{mean} corresponding to the point mean $= (X_{\text{last\_centre}} + X_{\text{best}})/2$ determined as pointed out in the previous section. If F_{mean} value is less than F_{best} value then, in given iteration, X displacement (or X movement) is computed and normalized twice: first each element of X is divided by the corresponding initial range (and thus the displacement is transformed into a unity-sided hypercube) and then that quantity is normalized again, dividing it by the diagonal of hypercube $\sqrt{m}$. These operations are illustrated as follows:

(1) normalized X_n (previous X for minimum):

$$X_n = \frac{(X - X_c)}{D}, \tag{6}$$

(2) normalized $X_{\min}$ (current X for minimum):

$$X_{\min n} = \frac{(X_{\min} - X_c)}{D}, \tag{7}$$

(3) normalized distance (should be bounded by 0 and sqrt of m):

$$d_n = \frac{\text{sum}\left((X_n - X_{\min n})^2\right)^{0.5}}{D}, \tag{8}$$

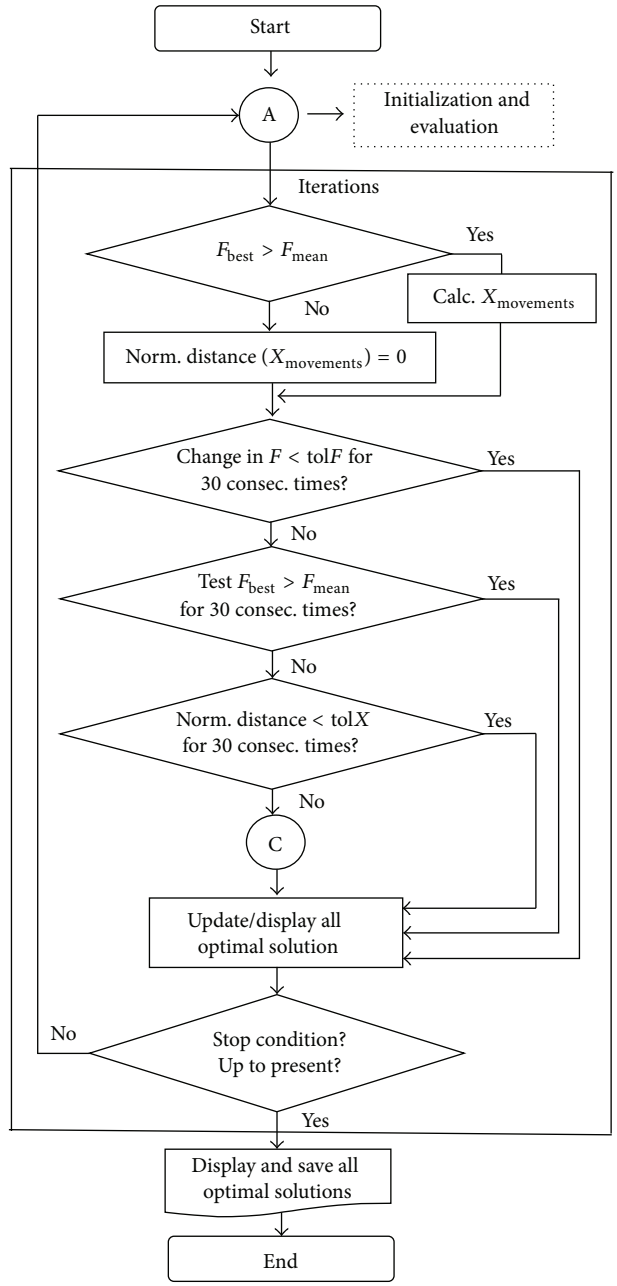


Figure 3: Flowchart of the displacement-shrink process. Here A is initialization and evaluation process and C is searching space process.

(4) renormalized distance (should be bounded by 0–0.1):

$$d_{nn} = \frac{d_n}{\sqrt{m}}. \tag{9}$$

In the result of these operations, X_n points are shrunk (become smaller) to the centre point X_c. These points are used to evaluate the test functions again. In the next blocks the hypercube continues moving and shrinking until one of the following conditions are not met.

(i) The change in consecutive F_{best} values is smaller than a preset value (tol$F$), for a preset consecutive number of times. This is also interpreted as convergence in F space.

(ii) The same or worse F value is found consecutively a preset number of times. This is interpreted as nonconvergence in F space.

(iii) The change in best X value (renormalized distance) is smaller than a preset value (tol$X$), for a preset consecutive number of times. This is interpreted as convergence in $\mathfrak{R}^m$ space. The whole process is repeated until specific termination conditions are satisfied.

(iv) The maximum number of iterations is reached: of course, in this case convergence is not guaranteed, as possibly lower values could be found with more iteration.

Each condition is tested for thirty consecutive times. If these conditions are not satisfied then the searching space process will be initialized.

We need to notice that the movement of X will not be performed if the F_{mean} value will be larger than F_{best} value. In such case, the searching space process will be initialized.

2.4. Searching Space Process. The searching space process initializes new center and size (radii) in order to create new hypercube. The objective function is evaluated at new points which are randomly chosen from the hypercube and having uniform distribution. The searching space process controls the movements of X according to the interval defined, in particularly for $X_{\text{movements}} < 0.1$. The value of X_{movement} is determined by d_{nn}. The flowchart of the searching space process of the HO algorithm is illustrated in Figure 4.

If the movement of X satisfies the condition then a factor of convergence S is calculated and updated at each iteration:

$$S = 1 - 0.2e^{-3d_{nn}}, \tag{10}$$

where d_{nn} is computed by (9) and describes the normalized distance moved by the average of last two best values of X. Next the update of solutions will be performed. The size (in all the dimensions) of the hypercube is reduced by multiplying by this factor. Thus, the hypercube reduces or maintains its size for nontrivial movements and shrinks otherwise. The whole process is repeated until specific termination conditions are satisfied.

3. Test Functions

The proposed hypercube optimization algorithm is tested on five continues test functions which are widely used in the literatures: *Ackley path function*, *Rastrigin function*, *Rosenbrock function*, *Griewank function,* and *Sphere function* [19–23]. The test functions are more applicable for the experimental evaluations of methods used in global optimization problems. The designed algorithm is implemented in MATLAB.

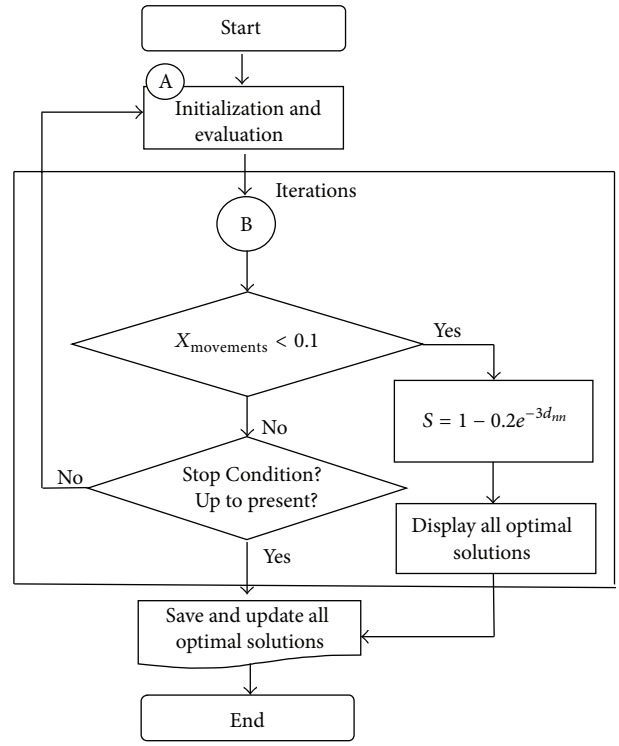


Figure 4: Flowchart of the searching space process. Here A is initialization and evaluation process and B is displacement-shrink process.

3.1. Ackley Path Function. *Ackley path function* is continuous, scalable, and nonseparable and is an extensively multimodal test function.

This test function is formulated as follows:

$$f_1(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e, \tag{11}$$

where D is a number of dimensions and $x_i = (x_1, x_2, \ldots, x_D)$ is D dimensional row vector. The test area is usually evaluated in the interval of $-32 \le x_i \le 32$, $i = (1, \ldots, D)$. Global minimum $f(x) = 0$ is obtainable for $x_i = (0, 0)$.

3.2. Rastrigin Function. *Rastrigin function* is continuous, scalable, and separable and is highly multimodal global optimization function.

This test function is formulated as follows:

$$f_2(x) = 10D + \sum_{i=1}^{D}\left(x_i^2 - 10\cos(2\pi x_i)\right), \tag{12}$$

where D is a number of dimensions and $x_i = (x_1, x_2, \ldots, x_D)$ is D dimensional row vector. The test area is usually evaluated in the interval of $-5.12 \le x_i \le 5.12$, $i = (1, \ldots, D)$. Global minimum $f(x) = 0$ is obtainable for $x_i = (0, 0)$.

3.3. Rosenbrock Valley Function. *Rosenbrock's valley function* is known as the *second function of De Jong*. This test function is continuous, scalable, naturally nonseparable, nonconvex, and unimodal.

This test function is formulated as follows:

$$f_3(x) = \sum_{i=1}^{D-1} \left[100\left(x_{i+1} - (x_i)^2\right)^2 + (x_i - 1)^2 \right], \quad (13)$$

where $\mathbf{D} \geq \mathbf{2}$ is a number of dimensions and $x_i = (x_1, x_2, \ldots, x_D)$ is D dimensional row vector. The test area is usually evaluated in the interval of $-2.048 \leq x_i \leq 2.048$, $i = (1, \ldots, D)$. Global minimum $f(x) = 0$ is obtainable for $x_i = (1, 1)$.

3.4. Sphere Function. The simplest benchmark function is *sphere model* which is also called De Jong's function 1. This test model is continuous, unimodal, and appearance of convex.

This test function is formulated as follows:

$$f_4(x) = \sum_{i=1}^{D} {x_i}^2, \quad (14)$$

where D is a number of dimensions and $x_i = (x_1, x_2, \ldots, x_D)$ is a dimensional row vector. The test area is usually evaluated in the interval of $-5.12 \leq x_i \leq 5.12$, $i = (1, \ldots, D)$. Global minimum $f(x) = 0$ is obtainable for $x_i = (0, 0)$.

3.5. Griewank Function. *Griewank function* is continuous, scalable, nonseparable, and multimodal test function.

This test function is formulated as follows:

$$f_5(x) = \frac{1}{4000} \sum_{i=1}^{D} {x_i}^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1, \quad (15)$$

where D is a number of dimensions and $x_i = (x_1, x_2, \ldots, x_D)$ is a dimensional row vector. The test area is usually evaluated in the interval of $-600 \leq x_i \leq 600$, $i = (1, \ldots, D)$. Global minimum $f(x) = 0$ is obtainable for $x_i = (0, 0)$.

4. Simulation Studies

The performance of the hypercube optimization algorithm is tested on the five benchmark functions given above. The benchmark functions $f_1 \div f_5$ are evaluated by considering the cases in which the problem dimensions are set as 1000, 5000, or even 10000 dimensions. At first the dimension is set as 1000. The population size is also set to 100, 1000, or even 10000. We have summarized the best average fitness (e.g., the lowest objective function value) and the average number of the test function evaluations over successful 30 runs. For each evaluation, the learning of the algorithm is continued 5000 iterations. The hypercube optimization algorithm has global minimum that was obtained with much well convergence process for these test functions.

No optimization algorithm guarantees convergence for any function, but it is a good practice to test the HO algorithm for several benchmark functions and tune the parameters.

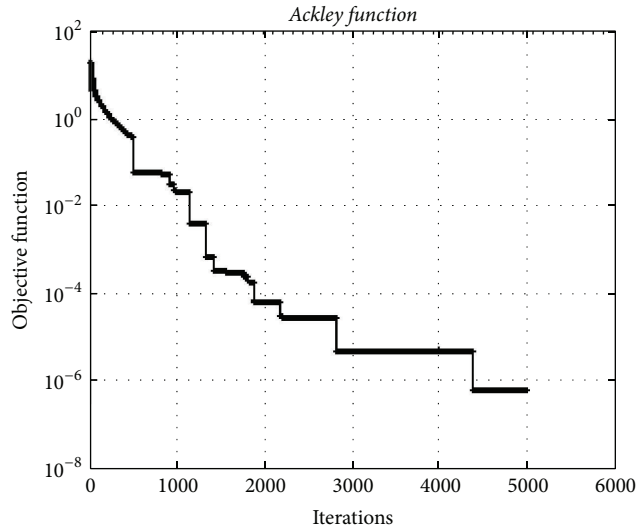


Figure 5: The convergence graphic for *the Ackley function* with dimension of 5000 and population of 100.

Therefore, we have tested the hypercube optimization algorithm on a set of benchmark functions, and the algorithm has yielded improved results, sometimes reaching the better solution faster than well-established algorithms. The details regarding the visualization of the test function results are given below.

In the next step, the test functions are evaluated for the cases in which the problem dimensions of $f_1 \div f_5$ are set to 5000 or even 10000 dimensions. The population size is set to 100. The convergence graphics have also been obtained and averaged through evaluations over successful 30 runs. The details of results regarding the visualization of the test function are given as follows.

4.1. Ackley Path Function. *The Ackley path function* is an extensively used multimodal test function. Figure 5 ilustrates the convergence graphic of HO algorithm for 5000 dimensions. The population size of the HO algorithm is almost insensitive to the dimension of the problems. The minimum of *Ackley* test function was obtained as $2.76e-07$.

Figure 6 depicts the convergence graphic of the HO algorithm for the *Ackley* test function having 10000 dimensions. The minimum value of the function was obtained as $1.16e-06$.

4.2. Rastrigin Function. The *Rastrigin function* is a typical nonlinear multimodal function. This test function is a fairly difficult problem for evolutinary algorithms due to the high number of dimensions and large number of local minima.

Figure 7 depicts the convergence graphic of HO algorithm for the *Rastrigin* test function having 5000 dimensions. The minimum was obtained as $7.13e-10$. The HO algorithm can find near-optimal solutions with much well convergence with high dimension for this test function.

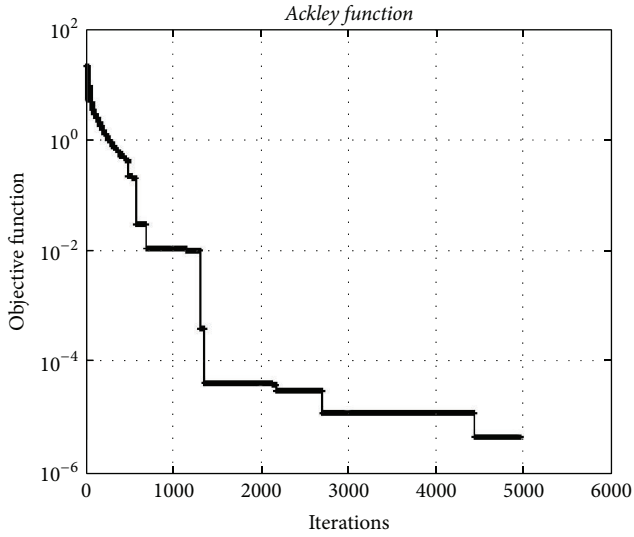


Figure 6: The convergence graphic for the *Ackley function* with dimension of 10000 and population of 100.

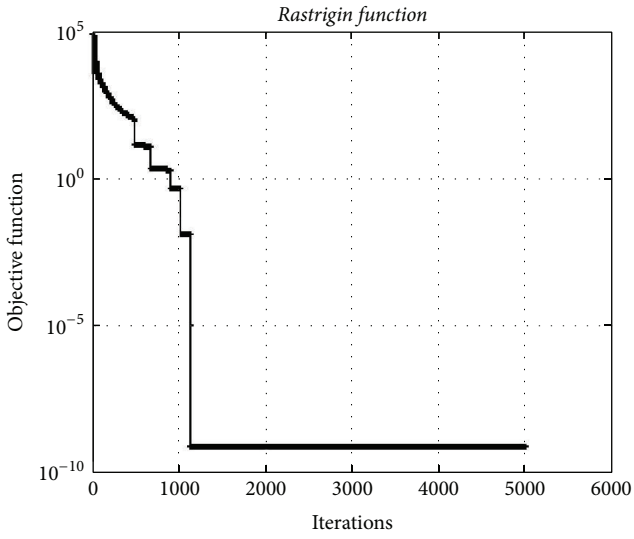


Figure 7: The convergence graphic for the *Rastrigin function* with dimension of 5000 and population of 100.

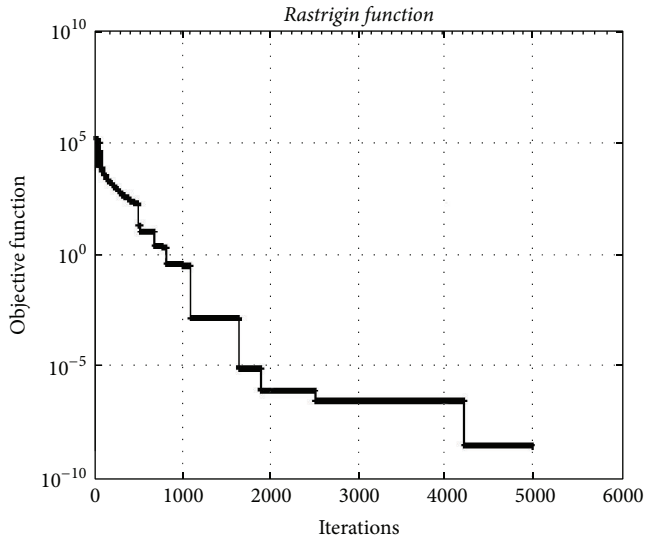


Figure 8: The convergence graphic for the *Rastrigin function* with dimension of 10000 and population of 100.

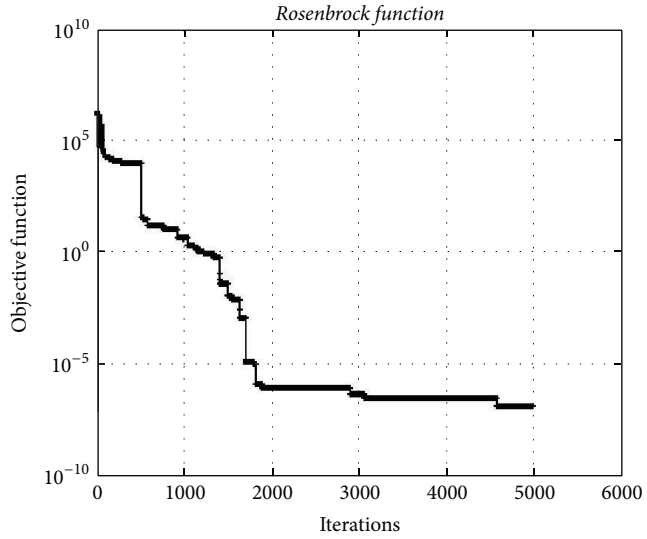


Figure 9: The convergence graphic for the *Rosenbrock function* having dimension of 5000 and population of 100.

Figure 8. It depicts the convergence graphic for the test function having 10000 dimensions. The minimum value of function was obtained as $2.99e-09$.

4.3. Rosenbrock Function. The *Rosenbrock function* is a typical naturally nonseparable, nonconvex, and unimodal. This test function is also a fairly hard problem for evolutionary algorithms.

Figure 9 depicts the convergence graphic for the *Rosenbrock test function* having 5000 dimensions. The minimum value of function was obtained as $1.15e-08$. The HO algorithm can find optimal or near-optimal solutions with much well convergence. This fact indicates that HO algorithm is almost insensitive to the dimension of the problems.

Figure 10 depicts the convergence graphic for the *Rosenbrock function* having 10000 dimensions. The minimum value of function was obtained as $3.38e-08$.

4.4. Sphere Function. The *Sphere function* is a typical unimodal test function. Figure 11 depicts convergence graphic of HO algorithm for the *Sphere* test function having 5000 dimensions. The minimum value of test function using HO algorithm was obtained as $4.64e-020$.

In Figure 12, the convergence graphic of hypercube optimization algorithm for the *Sphere* test function having 10000 dimensions is given. The minimum value was obtained as $2.40e-016$ with much well convergence. This test function

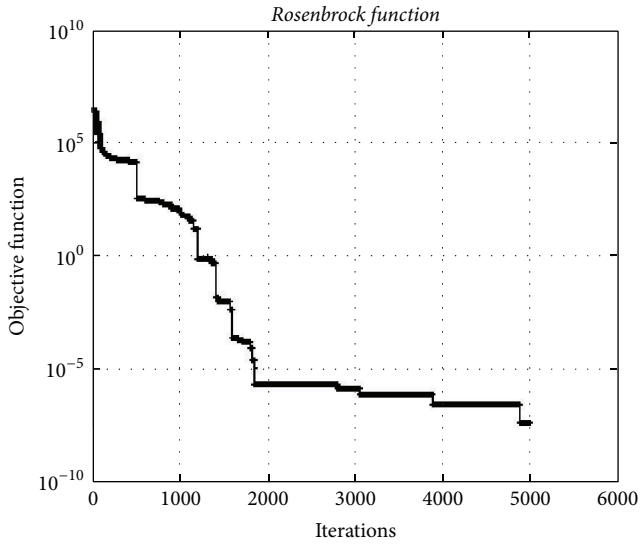


Figure 10: The convergence graphic for the *Rosenbrock function* having dimension of 10000 and population of 100.

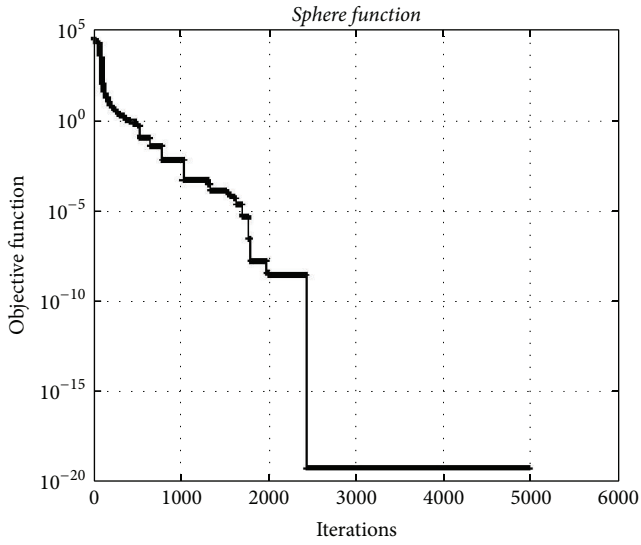


Figure 11: The convergence graphic for the *Sphere function* having 5000 dimensions and 100 populations.

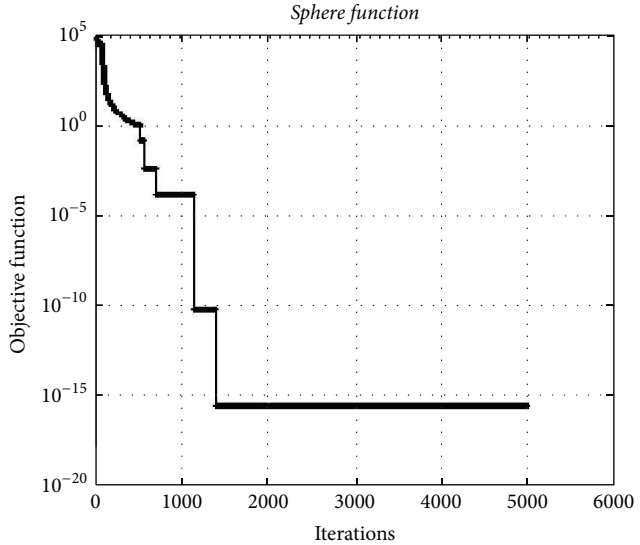


Figure 12: The convergence graphic for the *Sphere function* having 10000 dimensions and 100 populations.

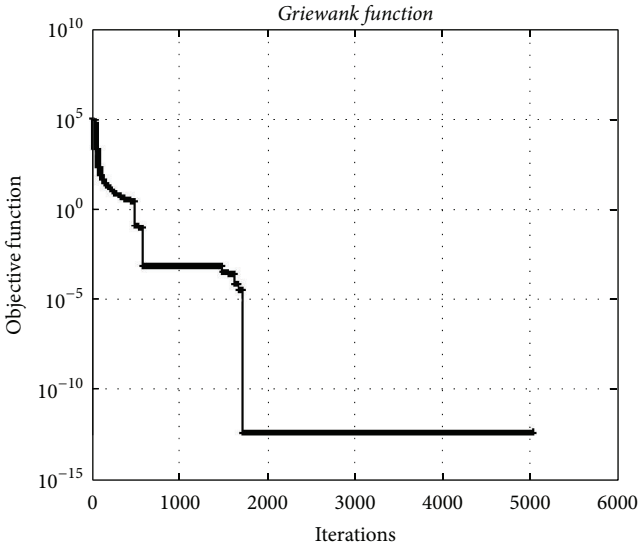


Figure 13: The convergence graphic for the *Griewank function* having dimension of 5000 and population of 100.

is a fairly easy problem for finding the total optimum and in the fast convergence.

4.5. Griewank Function. The *Griewank function* is also a typical nonlinear multimodal function. This test function is tested using many multiobjective evolutionary algorithms [23].

Figure 13 depicts the convergence graphic for the *Griewank* test function having 5000 dimensions. The minimum value of function was obtained as $3.34e-013$.

In Figure 14, the minimum value of test function using HO algorithm was obtained as $1.11e-016$ for 10000 dimensions. The HO algorithm can find optimal or near-optimal solutions with much well convergence with high dimension for this test function.

5. Comparison

The hypercube optimization algorithm has yielded in general quite better results, sometimes reaching the better solution faster than well-established algorithms. The usage of multiobjective evolutionary algorithms allows us to find global optimal solutions and avoid local optimum problem.

The simulation results of HO algorithm that was obtained with test functions with different dimensions and averaged over 30 runs are given in Table 2. Using the table we can see

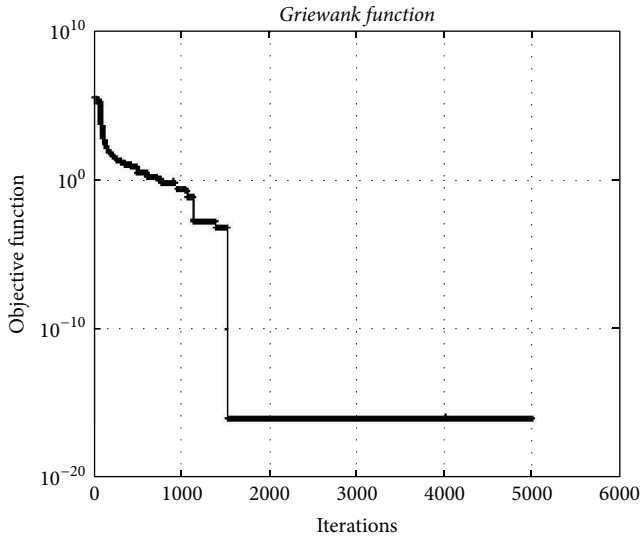


Figure 14: The convergence graphic for the *Griewank function* having dimension of 10000 and population of 100.

Table 2: Results of the mean best functions values averaged over 30 runs obtained by HO algorithm.

Function	Population	Dimension	The best	Iterations
	100	1000d	5.01$e-012$	5000
f_1	1000	1000d	2.46$e-013$	5000
	10000	1000d	5.12$e-012$	5000
	100	1000d	1.83$e-010$	5000
f_2	1000	1000d	4.54$e-011$	5000
	10000	1000d	3.63$e-011$	5000
	100	1000d	5.68$e-017$	5000
f_3	1000	1000d	8.16$e-017$	5000
	10000	1000d	2.06$e-017$	5000
	100	1000d	1.56$e-059$	5000
f_4	1000	1000d	5.86$e-059$	5000
	10000	1000d	1.12$e-072$	5000
	100	1000d	2.22$e-015$	5000
f_5	1000	1000d	6.32$e-015$	5000
	10000	1000d	5.44$e-015$	5000

that by increasing learning iterations from 1000 to 5000, the performance of HOA is increased for functions f_1, f_2, f_3, and f_4 as 2.46$e-013$, 4.54$e-011$, 8.16$e-017$, 6.32$e-015$, and 5.86$e-059$ correspondingly.

This chapter presents comparison of the performances of the hypercube optimization algorithm, with the two popular global optimization approaches, namely, genetic algorithm (GA) and particle swarm optimization (PSO) acting on above given four benchmark functions, namely, *Ackley path function*, *Rastrigin function*, *Rosenbrock function,* and *Sphere function*. These test functions are evaluated by considering the cases in which the problem dimensions of f_1, f_2, f_3, and f_4 are set as $D = 1000$ for the 1000 iterations. The proposed algorithm is tested by using above given test functions and the main unknown parameters are determined. The values of

Table 3: Comparison of the results.

Function	HO algorithm	GA	PSO	Iterations
f_1	1.07$e-003$	7.87	9.02	1000
f_2	6.07$e-004$	1.07$e+04$	1.40$e+04$	1000
f_3	5.13$e-002$	1.12$e+03$	6.58$e+06$	1000
f_4	1.16$e-008$	3.45$e+03$	5.50$e+03$	1000

main parameters for GA and PSO used in this chapter can be found in detail in [19, 29].

In Table 3, comparison of all the three algorithms for test functions of 1000 dimensions is provided.

All the algorithms were tested for 1000 dimensions. As evident from the results presented in Table 3, the HO algorithm obtains better results (reflected in the average fitness) than other techniques. The comparative results of the algorithms demonstrate that the performance of HO algorithm improves upon other well-known global optimization techniques: GA and PSO.

6. Conclusion

This paper proposes the hypercube optimization algorithm to solve multivariate systems for global optimization. The designed algorithm is based on a hypercube evaluation driven by convergence. The use of stochastic search method approach allows it to speed up the learning of the system and, respectively, to decrease training time of the system with a faster convergence. The simulations have been carried out using benchmarking functions, such as *Ackley function*, *Rastrigin function*, *Rosenbrock function*, *Sphere function,* and *Griewank function*. The computational results have demonstrated that the performance of the system have considerably increased in optimization problems for solving a set of global optimization problems with large numbers of populations. The population size of the HO algorithm is almost sensitive to the dimension of the problems for these test functions. The comparative results of HOA, GA, and PSO algorithms demonstrate that the performance of HO algorithm is an improvement upon other two global optimization techniques.

Conflict of Interests

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

References

[1] G. S. Fishman, *Monte Carlo: Concepts, Algorithms, and Applications*, Springer, New York, NY, USA, 1995.

[2] G. P. Lepage, "A new algorithm for adaptive multidimensional integration," *Journal of Computational Physics*, vol. 27, no. 2, pp. 192–203, 1978.

[3] D. J. Weiss and G. G. Kingsbury, "Application of computerized adaptive testing to educational problems," *Journal of Educational Measurement*, vol. 21, pp. 361–375, 1984.

[4] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Publishing, 1989.

[5] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Mich, USA, 1975.

[6] K. Deb, A. Anand, and D. Joshi, "A computationally efficient evolutionary algorithm for real-parameter optimization," *Evolutionary Computation*, vol. 10, no. 4, pp. 371–395, 2002.

[7] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[8] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, IEEE, Perth, Australia, December 1995.

[9] S. Kirkpatrick, J. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[10] F. Glover and M. Laguna, *Tabu Search*, Kluwer Academic Publishers, Boston, Mass, USA, 1997.

[11] M. Dorigo, *Optimization, learning and natural algorithms [Ph.D. thesis]*, Politecnico di Milano, Milano, Italy, 1992.

[12] D. Karaboga, "Artificial bee colony algorithm," *Scholarpedia*, vol. 5, no. 3, article 6915, 2010.

[13] R. H. Abiyev and M. Menekay, "Fuzzy portfolio selection using genetic algorithm," *Soft Computing*, vol. 11, no. 12, pp. 1157–1163, 2007.

[14] R. H. Abiyev, "Fuzzy wavelet neural network for prediction of electricity consumption," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM*, vol. 23, no. 2, pp. 109–118, 2009.

[15] T. Vidal, T. G. Crainic, M. Gendreau, N. Lahrichi, and W. Rei, "A hybrid genetic algorithm for multidepot and periodic vehicle routing problems," *Operations Research*, vol. 60, no. 3, pp. 611–624, 2012.

[16] X. Yang, C.-B. Hu, K.-X. Peng, and C.-N. Tong, "Load distribution of evolutionary algorithm for complex-process optimization based on differential evolutionary strategy in hot rolling process," *Mathematical Problems in Engineering*, vol. 2013, Article ID 675381, 8 pages, 2013.

[17] L. Zhang, M. Zhang, W. Yang, and D. Dong, "Golden ratio genetic algorithm based approach for modelling and analysis of the capacity expansion of urban road traffic network," *Computational Intelligence and Neuroscience*, vol. 2015, Article ID 512715, 9 pages, 2015.

[18] J.-T. Tsai, J.-H. Chou, and W.-H. Ho, "Improved quantum-inspired evolutionary algorithm for engineering design optimization," *Mathematical Problems in Engineering*, vol. 2012, Article ID 836597, 27 pages, 2012.

[19] C. Grosan and A. Abraham, "A novel global optimization technique for high dimensional functions," *International Journal of Intelligent Systems*, vol. 24, no. 4, pp. 421–440, 2009.

[20] Z. Yang, K. Tang, and X. Yao, "Differential evolution for high-dimensional function optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 3523–3530, IEEE, Singapore, September 2007.

[21] X. You, "Differential evolution with a new mutation operator for solving high dimensional continuous optimization problems," *Journal of Computational Information Systems*, vol. 6, no. 9, pp. 3033–3039, 2010.

[22] J. Brest, A. Zamuda, B. Bošković, M. S. Maučec, and V. Žumer, "High-dimensional real-parameter optimization using self-adaptive differential evolution algorithm with population size reduction," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '08)*, pp. 2032–2039, Hong Kong, June 2008.

[23] P. Korosec and J. Silc, "High-dimensional real-parameter optimization using the differential ant-stigmergy algorithm," *International Journal of Intelligent Computing and Cybernetics*, vol. 2, no. 1, pp. 34–51, 2009.

[24] J. J. Jamian, M. N. Abdullah, H. Mokhlis, M. W. Mustafa, and A. H. Bakar, "Global particle swarm optimization for high dimension numerical functions analysis," *Journal of Applied Mathematics*, vol. 2014, Article ID 329193, 14 pages, 2014.

[25] T. Hendtlass, "Particle swarm optimisation and high dimensional problem spaces," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 1988–1994, May 2009.

[26] I. Hamid, A. El-Bastawesy, I. Adel, M. Abdel-Salam, and M. Alaa El-Laffy, "A modified multi-scale particle swarm optimization for high dimensional systems," in *Proceedings of the 1st Taibah University International Conference on Computing and Information Technology (ICCIT '12)*, Al-Madinah Al-Munawwarah, Saudi Arabia, 2012.

[27] R. G. Regis, "Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 326–347, 2014.

[28] S. He, Q. H. Wu, and J. R. Saunders, "Group search optimizer: an optimization algorithm inspired by animal searching behavior," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 973–990, 2009.

[29] A.-R. Hedar and M. Fukushima, "Tabu search directed by direct search methods for nonlinear global optimization," *European Journal of Operational Research*, vol. 170, no. 2, pp. 329–349, 2006.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 638068, 7 pages
http://dx.doi.org/10.1155/2015/638068

Research Article

Phase Response Design of Recursive All-Pass Digital Filters Using a Modified PSO Algorithm

Wei-Der Chang

Department of Computer and Communication, Shu-Te University, Kaohsiung 824, Taiwan

Correspondence should be addressed to Wei-Der Chang; wdchang@stu.edu.tw

Received 5 October 2014; Revised 22 December 2014; Accepted 1 January 2015

Academic Editor: Rahib H. Abiyev

This paper develops a new design scheme for the phase response of an all-pass recursive digital filter. A variant of particle swarm optimization (PSO) algorithm will be utilized for solving this kind of filter design problem. It is here called the modified PSO (MPSO) algorithm in which another adjusting factor is more introduced in the velocity updating formula of the algorithm in order to improve the searching ability. In the proposed method, all of the designed filter coefficients are firstly collected to be a parameter vector and this vector is regarded as a particle of the algorithm. The MPSO with a modified velocity formula will force all particles into moving toward the optimal or near optimal solution by minimizing some defined objective function of the optimization problem. To show the effectiveness of the proposed method, two different kinds of linear phase response design examples are illustrated and the general PSO algorithm is compared as well. The obtained results show that the MPSO is superior to the general PSO for the phase response design of digital recursive all-pass filter.

1. Introduction

An all-pass filter means that its magnitude response is exactly equal to some constant value at all frequencies and independent of frequencies. The function of the all-pass filter is mainly to offer a phase modification without changing the magnitude on a given filter. It is rather useful in the theory of minimum-phase systems, in transforming frequency-selective low-pass filters into other frequency-selective forms, and in obtaining variable-cutoff frequency-selective filters [1]. Due to these advantages, the all-pass filter has been applied in many signal processing applications, including the group-delay equalization, complementary filter banks, multirate filtering, and other fields [2–8]. A large number of methods for designing all-pass filter have been developed in recent years. In [2], for example, the author proposed a new design method for an all-pass filter where it has a least squares or an equiripple phase-error response. It is based on formulating a weighted error between the desired and the actual phase responses in a quadratic form. Filter coefficients can be solved by using a Toeplitz-plus-Hankel matrix. In [3], an IIR all-pass filter with equiripple phase response was designed based on the eigenvalue problem and this design problem can be formulated as the representation of an eigenvalue problem via the Remez exchange algorithm. A Hopfield neural network was combined to the design of IIR all-pass digital filters [5]. In the case, filter coefficients can be evaluated by Hopfield neural networks in a parallelism manner in accordance with the error function that is formulated as a Lyapunov energy function. In addition, the authors developed a digital linear phase notch filter design scheme based on IIR all-pass filter. The designed filter can be realized by parallel connection of two IIR all-pass filters with approximately linear phase. Design algorithms exhibit fast convergence and easy initial values determination [7].

Unlike the above-mentioned design schemes, this paper attempts to utilize a modified particle optimization (MPSO) algorithm to solve the digital recursive all-pass filter design problem. This developed algorithm is a variant of the general PSO but it has a better searching capacity in solving optimized problems. The detailed description for such MPSO algorithm will be addressed later. The remainder of this paper is summarized as follows. Section 2 gives a brief description for the recursive all-pass digital filter. In Section 3, a modified

PSO algorithm is introduced in detail and the MPSO-based design steps for all-pass digital filter are also given. Section 4 will provide two different kinds of examples to confirm the applicability of the proposed method and some comparisons with the general PSO are further made. Finally, a conclusion about the proposed method is simply described in Section 5.

2. Recursive All-Pass Digital Filter

Let us consider a recursive all-pass digital filter whose transfer function is expressed by

$$H(z) = \frac{a_N + a_{N-1}z^{-1} + \cdots + a_0 z^{-N}}{a_0 + a_1 z^{-1} + \cdots + a_N z^{-N}} = \frac{z^{-N} D\left(z^{-1}\right)}{D(z)}, \quad (1)$$

where N represents the order of the filter, a_0 is always set to be 1, $a_1, a_2, \ldots, a_N$ are real coefficients, and $D(z) = \sum_{i=0}^{N} a_i z^{-i}$. Let $z = e^{j\Omega}$, where Ω denotes the digital frequency, substituting it into (1) to derive the frequency response. The following magnitude response can be easily obtained:

$$\begin{aligned} |H(\Omega)| &= \frac{\left|z^{-N} D\left(z^{-1}\right)\right|}{|D(z)|} = \frac{\left|D\left(z^{-1}\right)\right|}{|D(z)|} \\ &= \frac{\left|1 + a_1 e^{j\Omega} + \cdots + a_N e^{jN\Omega}\right|}{\left|1 + a_1 e^{-j\Omega} + \cdots + a_N e^{-jN\Omega}\right|} = 1. \end{aligned} \quad (2)$$

It is seen from (2) that the magnitude response is equal to one at all frequencies; that is, it is independent of the filter coefficients. Furthermore, its phase response is derived by

$$\begin{aligned} \theta(\Omega) &= -N\Omega + 2\arctan\left(\frac{\sum_{i=1}^{N} a_i \sin(i\Omega)}{\sum_{i=1}^{N} a_i \cos(i\Omega)}\right) \\ &= -N\Omega + 2\arctan\left(\frac{\mathbf{a}\mathbf{s}^T(\Omega)}{1 + \mathbf{a}\mathbf{c}^T(\Omega)}\right), \end{aligned} \quad (3)$$

where $\mathbf{s}(\Omega) \equiv [\sin(\Omega), \sin(2\Omega), \ldots, \sin(N\Omega)]$, $\mathbf{c}(\Omega) \equiv [\cos(\Omega), \cos(2\Omega), \ldots, \cos(N\Omega)]$, and

$$\mathbf{a} \equiv [a_1, a_2, \ldots, a_N] \quad (4)$$

is a parameter vector consisting of all filter coefficients. This vector fully dominates the phase response behavior of the digital filter. In this paper, we want to design the parameter vector $\mathbf{a}$ such that the phase response achieves certain design specification. Moreover, this vector $\mathbf{a}$ is called the particle or individual of the PSO algorithm and many such particles then form a population. Some adjusting mechanisms are utilized on the full population. Moreover, it can be easily seen from (3) that a highly complicated nonlinear function arctan(·) is involved and it is difficult to solve. Thus, (3) always needs to be modified as another form for the phase response design [3–5]. However, the proposed method in this paper can directly use (3) for the phase response design of recursive all-pass filter.

3. Modified Particle Swarm Optimization (MPSO) Algorithm

Kennedy and Eberhart initially proposed the PSO algorithm in 1995 and recently it became one of the popular and efficient optimization algorithms [9]. Like most swarm intelligence algorithms, PSO is also a population-based search algorithm. It simulates the social behavior of organisms, such as fish schooling and bird flocking. Each fish or bird, viewed as a particle or an individual, represents a candidate solution to the optimized problem. By the velocity and position updating formulas, each particle moves through the search space toward the global solution. Based on the PSO algorithm, various engineering optimization applications have been successively developed and explored in recent years, such as power system stabilizer design [10], PID controller design [11, 12], FPGA implementations [13, 14], Volterra filter modeling [15], QRD-based multirelay system design [16], automatic clustering [17], multifault classification [18], and aeroengine nonlinear programming model [19]. Besides, the authors developed a novel PSO algorithm in which the inertia weight is modified to enhance its search capability [20]. The proposed method has successfully been applied in the high pass FIR digital filter design. Another design method for the low pass FIR digital filter with linear phase properties was also developed [21]. A new definition for the velocity vector and swarm updating of the PSO algorithm was proposed.

At the beginning, PSO algorithm requires an objective function to judge the performance of the particle and also to guide the search direction of the algorithm. To solve the phase response design problem for the recursive all-pass digital filter, the objective function (OF) is defined by

$$\mathrm{OF} = \int_{\Omega_{\min}}^{\Omega_{\max}} \left|\theta_d(\Omega) - \theta(\Omega)\right| d\Omega, \quad (5)$$

where θ_d is the desired phase response given by the designer, θ is the actual phase response of the all-pass digital filter as described by (3), and $\Omega_{\min}$ and $\Omega_{\max}$ are the integral lower and upper bounds, respectively. The algorithm is utilized to minimize this objective function OF to achieve the optimal phase response design. Each particle is changed according to the following velocity formula of (6) and position formula of (7) for original PSO algorithm:

$$\begin{aligned} v_{ij}(n+1) &= w v_{ij}(n) + c_1 r_1 \left(p_{ij}(n) - a_{ij}(n)\right) \\ &\quad + c_2 r_2 \left(g_j(n) - a_{ij}(n)\right), \end{aligned} \quad (6)$$

$$a_{ij}(n+1) = a_{ij}(n) + v_{ij}(n+1), \quad (7)$$

where n denotes the nth iteration of the algorithm, v_{ij}, a_{ij}, and p_{ij} represent the velocity, position, and individual best position for the ith particle with respect to the jth dimension, respectively, g_j represents the global best position with respect to the jth dimension among the population, w is called the inertia weight, c_1 and c_2 are two positive acceleration coefficients that pull each particle toward the individual best and the global best positions, respectively, and r_1 and r_2 are two uniform distribution random numbers chosen from the interval [0, 1]. The PSO algorithm uses these two updating mechanisms to achieve the optimization.

In this study, a modified PSO (MPSO) algorithm is taken into the phase response design of recursive all-pass

digital filter [11, 15]. The difference between the original and modification is to change the velocity formula. In the MPSO, the population needs to be further divided into some subpopulations at the beginning; for example, suppose that the initial population includes 50 particles and it will be divided into five subpopulations. Thus, the first subpopulation is composed of particles from number one to number ten, and the second then contains particles from number eleven to number twenty, and so forth. The best particle of each subpopulation needs to be recorded according to its objective function. Instead of the velocity formula of (6), the MPSO algorithm uses the following improved version:

$$v_{ij}(n+1) = wv_{ij}(n) + c_1 r_1 \left(p_{ij}(n) - a_{ij}(n)\right) + c_2 r_2 \left(g_j(n) - a_{ij}(n)\right) + c_3 r_3 \left(s_j(n) - a_{ij}(n)\right), \quad (8)$$

where s_j is a new variable called the local best and represents the position of the best particle of the subpopulation where the ith particle is located, c_3 is also a positive acceleration coefficient, and r_3 is a random number selected from the range $[0, 1]$ uniformly.

Design steps of MPSO-based for the phase response design of the recursive all-pass digital filter can be summarized in the following.

Data. Filter order N in (1) and (3), desired phase response θ_d and integral lower and upper bounds $\Omega_{\min}$ and $\Omega_{\max}$ in (5), number of particles (population size) Ps, number of subpopulations S, number of generations G, inertia weight w, and positive constants c_1, c_2, and c_3 in (8).

Goal. Derive a recursive all-pass digital filter with the phase response approaching the desired response θ_d.

(1) Create an initial population consisting of Ps particles from the interval $[-1, 1]$ randomly.

(2) Divide the population into S subpopulations by particle serial numbers.

(3) If a prescribed number G of iterations are achieved, then the algorithm stops.

(4) Evaluate the objective function of (5) for each particle and record the related individual best, local best, and global best positions.

(5) Update each particle's velocity and position using (8) and (7).

(6) Go back to Step (3).

4. Simulation Results

In this section, we consider two different examples with linear phase design to show the applicability of our proposed method [2, 5]. Some comparisons with the general PSO are also performed. In the PSO and MPSO, the variables of the algorithm are given by $w = 0.8$, $c_1 = c_2 = 0.5$, and $w = 0.8$, $c_1 = c_2 = c_3 = 0.5$, respectively, for all of the following simulations.

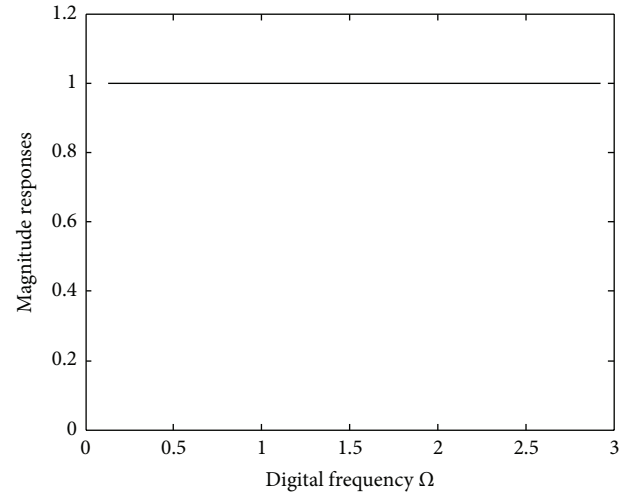


Figure 1: Magnitude response of Example 1.

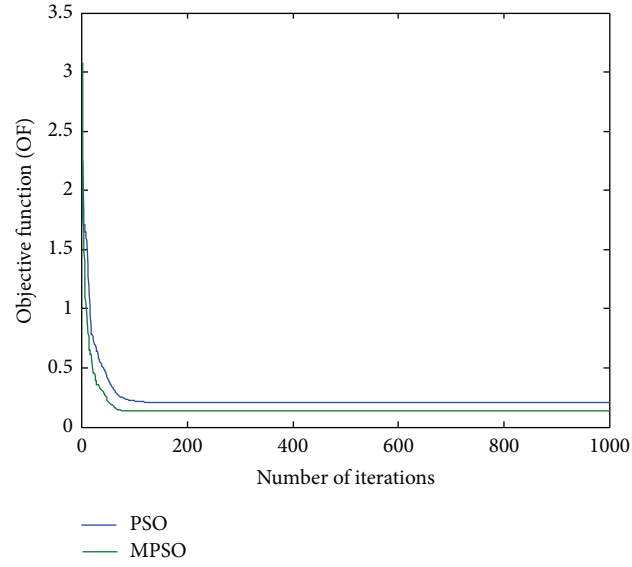


Figure 2: Trajectories of objective function (OF) for Run 1 of Example 1.

Example 1. In this example, the recursive all-pass filter is designed to approximate a desired Hilbert transformer whose phase response is given by

$$\theta_d(\Omega) = -N\Omega - \frac{\pi}{2}, \quad \Omega_{\min} \le \Omega \le \Omega_{\max}, \quad (9)$$

where the lower bound and upper bound are set to $\Omega_{\min} = 0.04\pi$ and $\Omega_{\max} = 0.94\pi$, respectively; N means the filter order and here it is chosen by $N = 10$. The magnitude response of such a recursive all-pass filter is plotted in Figure 1. In addition, the population size and number of generations are given by Ps = 20 and $G = 1000$ for the PSO and MPSO algorithm, and the number of subpopulations is simply set to $S = 4$ only for the MPSO. To verify the algorithm's robustness and efficiency, 20 independent runs with different initial conditions are executed for both

Table 1: Objective function values evaluated for Example 1.

	PSO algorithm	MPSO algorithm
Run 1	0.20570301	0.13606212
Run 2	0.13627085	0.13620043
Run 3	0.35740401	0.13679522
Run 4	0.13738652	0.13650124
Run 5	0.13672965	0.13680452
Run 6	0.14521057	0.13601797
Run 7	0.13619628	0.13673623
Run 8	0.15390549	0.13606745
Run 9	0.13755478	0.13762322
Run 10	0.14286368	0.13602069
Run 11	0.14405732	0.16636761
Run 12	0.14924150	0.13679615
Run 13	0.13622616	0.13703487
Run 14	0.13707200	0.13680303
Run 15	0.13634390	0.13701766
Run 16	0.21968176	0.22009637
Run 17	0.14488212	0.13649137
Run 18	0.14857613	0.13703097
Run 19	0.14250406	0.13624831
Run 20	0.27376677	0.14515192
Mean	0.16607883	0.14269337
Variance	0.00313722	0.00035958

Table 2: Digital filter coefficients derived by Run 1 of Example 1.

Filter coefficients	PSO algorithm	MPSO algorithm
a_1	−0.974	−0.9853
a_2	0.4350	0.4685
a_3	−0.3959	−0.4388
a_4	0.2323	0.2959
a_5	−0.1912	−0.2625
a_6	0.1048	0.1830
a_7	−0.0743	−0.1485
a_8	0.0268	0.0975
a_9	−0.0153	−0.0646
a_{10}	−0.0106	0.0290

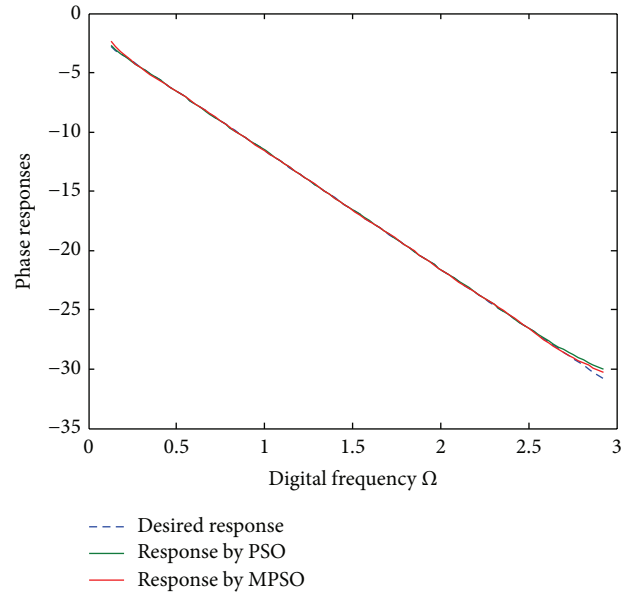


Figure 3: Phase responses for Run 1 of Example 1.

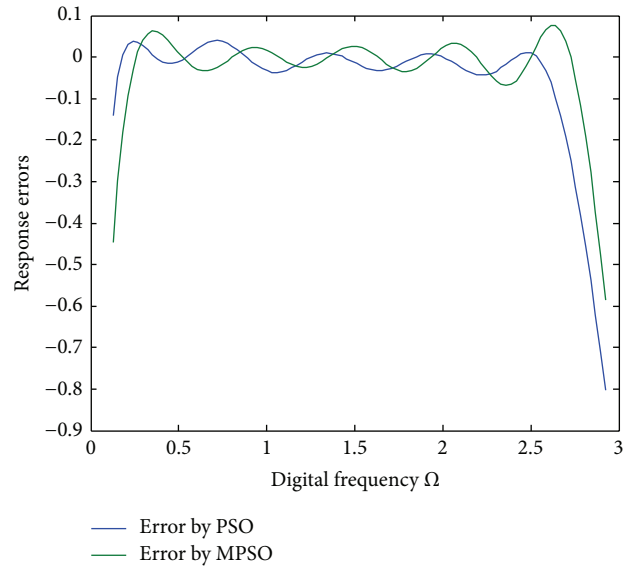


Figure 4: Phase response errors for Run 1 of Example 1.

algorithms. Final simulation results are listed in Tables 1 and 2 and shown in Figures 2–4, respectively. Table 1 lists the objective function values of 20 independent runs and it clearly reveals that the results by the MPSO are better than those by the PSO for most of independent runs. The mean and variance are evaluated by $u = 0.14269337$ and $\sigma^2 = 0.00035958$ for the MPSO and $u = 0.16607883$ and $\sigma^2 = 0.00313722$ for the PSO, respectively. To show the design outcomes, Figure 2 displays the convergence trajectories of the objective function for Run 1 of the proposed MPSO and PSO algorithms. As can be seen from Figure 2, the MPSO algorithm has a quicker convergence and lower objective function value than the PSO algorithm. Both phase responses and errors are further shown in Figures 3 and 4, respectively. A better simulation result can be obtained by the proposed method. In addition, all of digital filter coefficients derived by Run 1 of the PSO and MPSO algorithm are listed in Table 2 for comparisons.

Example 2. This example will design a recursive all-pass digital filter with a desired sinusoidal phase response expressed by

$$\theta_d(\Omega) = 4\pi(\cos\Omega - 1) - 52\Omega, \quad \Omega_{\min} \le \Omega \le \Omega_{\max}, \quad (10)$$

where $\Omega_{\min} = 0$ and $\Omega_{\max} = \pi$ are given. Its corresponding magnitude response is shown in Figure 5. In the simulation, a digital recursive filter with $N = 60$ is adopted and the population size and iterative number of the algorithms are

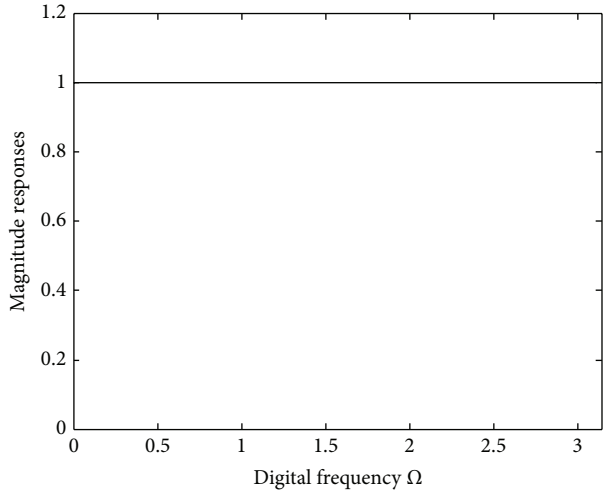


Figure 5: Magnitude response of Example 2.

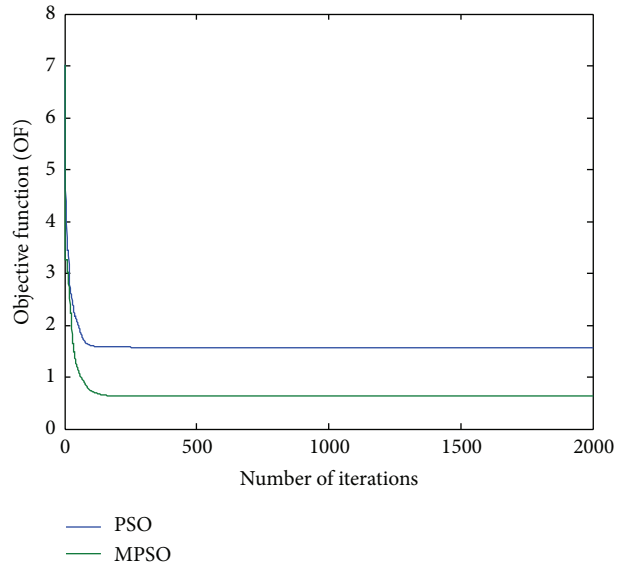


Figure 6: Trajectories of objective function (OF) for Run 1 of Example 2.

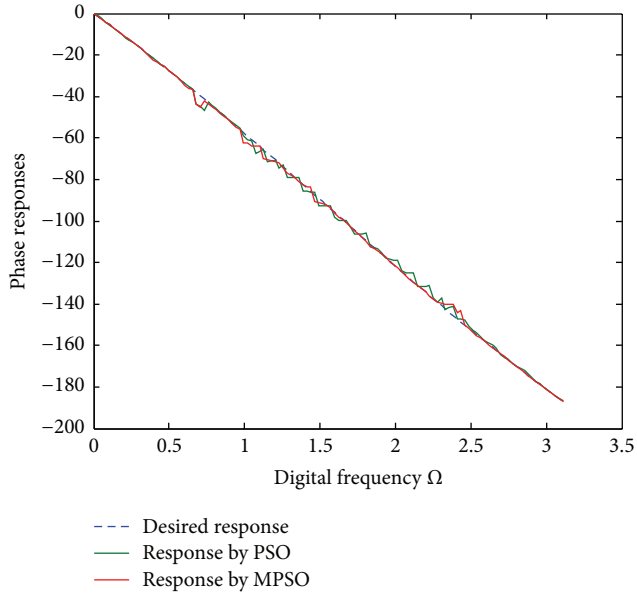


Figure 7: Phase responses for Run 1 of Example 2.

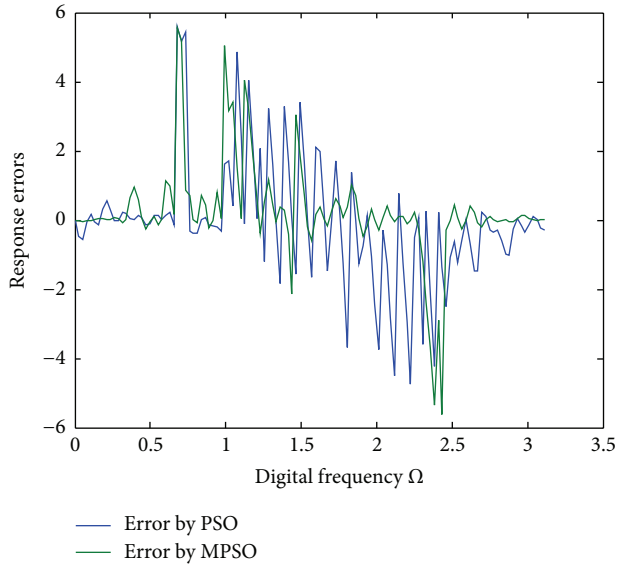


Figure 8: Phase response errors for Run 1 of Example 2.

set to Ps = 40 and $G = 2000$, respectively, for solving such a higher-order digital filter. Moreover, as given in Example 1, the number of subpopulations is chosen by $S = 4$ and 20 independent runs with different sets of initial conditions are also performed for certifying the robustness of the algorithm. Table 3 lists a comparison of the objective function values evaluated by the proposed MPSO and PSO for Run 1 to Run 20, respectively. Some of digital filter coefficients derived are listed in Table 4 for comparisons. Figures 6–8 then show the related design outcomes only for Run 1 of the PSO and proposed algorithm. Again, it can be concluded from these results that the proposed MPSO is superior to the general PSO in the phase response design of recursive all-pass digital filter.

5. Conclusions

This paper has developed a new design method for the phase response design of recursive all-pass digital filter. A modified PSO (MPSO) algorithm is suggested to design the filter coefficients such that the obtained phase response can approximate the desired response that is given previously. The difference between the MPSO and PSO is to modify the velocity updating formula of the algorithm. To improve the search capacity, a new factor of local-best particle for each subpopulation is introduced in the modified velocity formula. Finally, two different kinds of examples have been illustrated to verify the efficiency of the proposed method as compared

Table 3: Objective function values evaluated for Example 2.

	PSO algorithm	MPSO algorithm
Run 1	1.56939914	0.62645705
Run 2	1.85475888	0.57827778
Run 3	1.55263227	0.67573844
Run 4	1.57989126	0.67840205
Run 5	2.03270089	0.75624957
Run 6	1.63204094	0.67220282
Run 7	1.34828024	0.52125315
Run 8	1.65465543	1.18349923
Run 9	1.39729362	0.86232224
Run 10	1.55541996	0.65710581
Run 11	1.82677089	1.14715629
Run 12	1.68802195	0.62355711
Run 13	1.94169732	0.65469452
Run 14	1.61249872	0.48998564
Run 15	1.78693672	1.17072011
Run 16	1.75210045	1.02655545
Run 17	1.57667970	0.76927394
Run 18	1.64655678	0.80753180
Run 19	1.84461988	1.10135973
Run 20	1.56655437	0.44814752
Mean	1.67097547	0.77252451
Variance	0.02816276	0.05174895

Table 4: Digital filter coefficients derived by Run 1 of Example 2.

Filter coefficients	PSO algorithm	MPSO algorithm
a_1	0.0464	−0.1479
a_2	0.8023	1.1317
a_3	0.3584	0.0022
a_4	0.8848	0.8390
a_5	0.0292	−0.0636
a_6	0.4525	0.5609
⋮	⋮	⋮
a_{58}	0.0093	0.0089
a_{59}	−0.1014	0.1059
a_{60}	0.1847	0.0181

with the general PSO algorithm. Simulation results have sufficiently proven that the proposed MPSO has a better design outcome than the PSO in the phase response design of recursive all-pass digital filter.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was partially supported by the Ministry of Science and Technology of Taiwan under Grant MOST 103-2221-E-366-004.

References

[1] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*, Prentice-Hall, London, UK, 2nd edition, 1999.

[2] S. S. Kidambi, "Weighted least-squares design of recursive allpass filters," *IEEE Transactions on Signal Processing*, vol. 44, no. 6, pp. 1553–1557, 1996.

[3] X. Zhang, "Design of IIR digital allpass filters based on eigenvalue problem," *IEEE Transactions on Signal Processing*, vol. 47, no. 2, pp. 554–559, 1999.

[4] M. Lang, "Allpass filter design and applications," *IEEE Transactions on Signal Processing*, vol. 46, no. 9, pp. 2505–2514, 1998.

[5] L. C. Su, Y. D. Jou, F. K. Chen, and C. M. Sun, "Neural type least squares design for IIR all-pass filters," in *Proceedings of the National Symposium on Telecommunications*, pp. 147–150, Taipei, Taiwan, 2007.

[6] J.-W. Horng, "Current conveyors based allpass filters and quadrature oscillators employing grounded capacitors and resistors," *Computers and Electrical Engineering*, vol. 31, no. 1, pp. 81–92, 2005.

[7] G. Stančić and S. Nikolić, "Digital linear phase notch filter design based on IIR all-pass filter application," *Digital Signal Processing*, vol. 23, no. 3, pp. 1065–1069, 2013.

[8] W. D. Chang, S. T. Pan, K. H. Cheng, and M. C. Hsu, "Optimal design of allpass digital filters using artificial bee colony," in *Proceedings of the Workshop on Synthesis and System Integration of Mixed Information Technologies*, pp. 159–162, Beppu, Japan, 2012.

[9] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, Perth, Australia, November-December 1995.

[10] M. A. Abido, "Optimal design of power-system stabilizers using particle swarm optimization," *IEEE Transactions on Energy Conversion*, vol. 17, no. 3, pp. 406–413, 2002.

[11] W.-D. Chang and S.-P. Shih, "PID controller design of nonlinear systems using an improved particle swarm optimization approach," *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, no. 11, pp. 3632–3639, 2010.

[12] J.-S. Chiou, S.-H. Tsai, and M.-T. Liu, "A PSO-based adaptive fuzzy PID-controllers," *Simulation Modelling Practice and Theory*, vol. 26, pp. 49–59, 2012.

[13] M. A. Cavuslu, C. Karakuzu, and F. Karakaya, "Neural identification of dynamic systems on FPGA with improved PSO learning," *Applied Soft Computing Journal*, vol. 12, no. 9, pp. 2707–2718, 2012.

[14] B. Vasumathi and S. Moorthi, "Implementation of hybrid ANNPSO algorithm on FPGA for harmonic estimation," *Engineering Applications of Artificial Intelligence*, vol. 25, no. 3, pp. 476–483, 2012.

[15] W.-D. Chang, "Volterra filter modeling of nonlinear discrete-time system using improved particle swarm optimization," *Digital Signal Processing*, vol. 22, no. 6, pp. 1056–1062, 2012.

[16] C. Zhang, M. Wu, and L. Luan, "An optimal PSO distributed precoding algorithm in QRD-based multi-relay system," *Future Generation Computer Systems*, vol. 29, no. 1, pp. 107–113, 2013.

[17] S. J. Nanda and G. Panda, "Automatic clustering algorithm based on multi-objective Immunized PSO to classify actions of 3D human models," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 5-6, pp. 1429–1441, 2013.

[18] Z. Liu, H. Cao, X. Chen, Z. He, and Z. Shen, "Multi-fault classification based on wavelet SVM with PSO algorithm to analyze vibration signals from rolling element bearings," *Neurocomputing*, vol. 99, pp. 399–410, 2013.

[19] Y. Wang, B. Li, T. Sun, K. Jiang, and X. Wang, "A modified PSO algorithm for aero-engine non-linear programming model," in *Proceedings of the 8th World Congress on Intelligent Control and Automation (WCICA '10)*, pp. 4069–4073, Jinan, China, July 2010.

[20] S. Mondal, D. Chakraborty, R. Kar, D. Mandal, and S. P. Ghoshal, "Novel particle swarm optimization for high pass FIR filter design," in *Proceedings of the IEEE Symposium on Humanities, Science and Engineering Research (SHUSER '12)*, pp. 413–418, June 2012.

[21] S. Mukherjee, R. Kar, D. Mandal, S. Mondal, and S. P. Ghoshal, "Linear phase low pass FIR filter design using improved particle swarm optimization," in *Proceedings of the IEEE Student Conference on Research and Development (SCOReD '11)*, pp. 358–363, Cyberjaya, Malaysia, December 2011.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 780352, 13 pages
http://dx.doi.org/10.1155/2015/780352

Research Article

A Multiuser Manufacturing Resource Service Composition Method Based on the Bees Algorithm

Yongquan Xie,[1,2,3] Zude Zhou,[1,2] Duc Truong Pham,[3] Wenjun Xu,[1,2] and Chunqian Ji[3]

[1] *School of Information Engineering, Wuhan University of Technology, Luoshi Road 122, Wuhan 430070, China*
[2] *Key Laboratory of Fiber Optic Sensing Technology and Information Processing, Ministry of Education, Wuhan University of Technology, Wuhan 430070, China*
[3] *School of Mechanical Engineering, University of Birmingham, Edgbaston, Birmingham B152TT, UK*

Correspondence should be addressed to Yongquan Xie; xyqwhut0728@126.com

Received 25 December 2014; Accepted 25 May 2015

Academic Editor: Rafik Aliyev

In order to realize an optimal resource service allocation in current open and service-oriented manufacturing model, multiuser resource service composition (RSC) is modeled as a combinational and constrained multiobjective problem. The model takes into account both subjective and objective quality of service (QoS) properties as representatives to evaluate a solution. The QoS properties aggregation and evaluation techniques are based on existing researches. The basic Bees Algorithm is tailored for finding a near optimal solution to the model, since the basic version is only proposed to find a desired solution in continuous domain and thus not suitable for solving the problem modeled in our study. Particular rules are designed for handling the constraints and finding Pareto optimality. In addition, the established model introduces a trusted service set to each user so that the algorithm could start by searching in the neighbor of more reliable service chains (known as seeds) than those randomly generated. The advantages of these techniques are validated by experiments in terms of success rate, searching speed, ability of avoiding ingenuity, and so forth. The results demonstrate the effectiveness of the proposed method in handling multiuser RSC problems.

1. Introduction

Advanced manufacturing technologies or models, such as manufacturing grid (MGrid), global manufacturing (GM), virtual manufacturing (VM), agile manufacturing (AM), and cloud manufacturing (CMfg), have been put forward in order to respond to the rapidly changing market and enhance enterprise competitiveness. The more recent manufacturing models tend to adopt a service-oriented architecture, which can be implemented by using many interoperable standards. This architecture places an emphasis on knowledge and services and aims at realizing full sharing and circulation, high utilization, and on-demand use of various manufacturing resources and capabilities by providing safe, reliable, high quality, cheap, and on-demand used manufacturing services for the whole manufacturing lifecycle [1]. However, in the era of globalization, manufacturing resources are usually characterized by heterogeneously distributed in diverse systems and existed in multiple locations all over the world. Furthermore, these models nowadays incline to accept more open and transparent structures, where a service demander in one system may be a service provider in another system. Therefore the resource service allocation management that is capable of responding to any manufacturing task request rapidly, agilely, and accurately is a critical technique for the current manufacturing models and the next generation.

The shift from production-oriented manufacturing to service-oriented manufacturing necessitates a reform in the methods of resource service composition (RSC). As a key component of resource service allocation management, it takes advantage of existing resource services and integrates them into composite service that can be invoked to fulfill multiple complex manufacturing tasks beyond the capability

of any of the services performing singly. The research on RSC in manufacturing environment has increasingly attracted scholars. Some traditional methods for RSC mainly deal with internal manufacturing resources in an enterprise or some in collaborations, or they are unable to handle large-scale RSC optimal selection flexibly and intelligently. Other limitations of traditional RSC methods becoming apparent with the development of new market demands can be reflected as follows. (1) The majority of them strive to accomplish only one task request at a time by providing only one composite service, while ignoring the possibility that multiuser would publish their task requests simultaneously. Consequently the equilibrium allocation of resource services between multiple task requests has been neglected. For instance, resources of high quality may be allocated to a user only because he publishes his task request earlier than others but whose actual requirement is not stringent. (2) Mostly they fail to take into account the diverse situations of the request-and-response manufacturing environment. For example, they may reach a dead end when there are barely sufficient services available or only services of inferior quality available during a certain period but users still launch manufacturing task requests, or they are not competent enough to tackle very demanding task requests for special purposes. (3) Existing researches on RSC have emphasized on trust aggregation and QoS properties evaluation, respectively, for the improvement of RSC, but few of them have integrated the two aspects into one model. In an open manufacturing model a user may get involved in the whole lifecycle of manufacturing; therefore both the subjective properties like trust and objective properties like time and cost play a very important role for the user to make judgment and decisions to select services.

In consideration of the above insufficiencies, this study is to further optimally allocate manufacturing resource service, enhance resource utilization, and promote service composite efficiency. This paper investigates the RSC in a multiuser manufacturing environment, which is a constrained multiobjective combinational optimization problem and has shown to be NP-hard. Both objective and subjective QoS component are involved as objective functions or constraints in the model. The Bees Algorithm as a swarm-based intelligent algorithm is made adapted to solve the problem. Experiments are conducted to look into the performance of the proposed method based on a few criteria such as success rate, fraud service handling, and search speed. How the amended algorithm reacts to the diverse task requests and how its colony size affects the RSC performance are analyzed.

The rest of the paper is organized as follows. The next section introduces the preliminary knowledge and reviews related works on QoS properties evaluation, methods of RSC optimal selection, and the basic Bees Algorithm. Section 3 proposes a multiuser RSC model based on existing trust aggregation and QoS evaluation methods. This is followed by a detailed description of the Bees Algorithm suitable for solving the problem modeled in Section 4. The representation scheme of a solution, the probabilistic neighborhood search and shrinking strategy, the rules for handling multiple objective functions and constraints, and the time complexity of the algorithm are elaborated. In Section 5, the effectiveness of the proposed method is demonstrated and analyzed by groups of experiments. Finally, Section 6 summarizes the paper and draws directions for further researches.

Table 1: The aggregation of QoS properties from composite structures.

Properties	Sequential	Parallel	Conditional	Loop
Additive	$\sum_{i=1}^{n} q(s_i)$	$\sum_{i=1}^{n} q(s_i)$	$\sum_{i=1}^{n} q(s_i) \cdot p_i$	$n \cdot q(s_i)$
Multiplicative	$\prod_{i=1}^{n} q(s_i)$	$\prod_{i=1}^{n} q(s_i)$	$\sum_{i=1}^{n} q(s_i) \cdot p_i$	$q(s_i)^n$
Max-operator	$\sum_{i=1}^{n} q(s_i)$	$\max\{q(s_i)\}$	$\sum_{i=1}^{n} q(s_i) \cdot p_i$	$n \cdot q(s_i)$

2. Preliminary Knowledge and Related Works

2.1. QoS and Trust Aware RSC Optimal Selection. As the number of resource services of similar functionality registered in the service pool increase, users are more concerned about their nonfunctional properties. The functionality refers to what task a service can finish, while the nonfunctional properties are more aware of the QoS such as price, raw material cost, maintainability, and response time. A stand-alone service is often limited in its functionality and a manufacturing task usually demands multiple services in combination as a composite service chain. The technique of evaluating the QoS of a composite service has been investigated for years [2, 3]. Generally, existing researches derive the aggregated QoS from stand-alone services, making the interconnection between them quite significant. The most commonly used composite service structures encompass Sequential, Parallel, Conditional and Loop. The QoS properties are differentiated into three categories: additive, multiplicative, and max-operator. The aggregations of these properties in different categories are given in Table 1 [4]. In the table, $q(s_i)$ denotes the value of corresponding QoS property of the component service s_i. These values should be normalized to the same scale according to Formula (1) in order to avoid inaccuracy and incommensurability due to the distinctive measurement metrics that different QoS properties adopt [5]. In Formula (1), the property value to be maximized is said to be positive, and the one to be minimized is negative. Complicated composite service structures can be simplified by mature techniques, such as transforming to sequential structures [6, 7]. Then the relationship between local and global constraints can be deduced [4]. Apart from this bottom-up method for handling QoS properties, a top-down method is also studied [8], which is claimed to be efficient to address the global constraints, especially for dynamic and adaptable composition of services.

As to the QoS-ware RSC optimal selection, methods and algorithms are constantly being developed. A composition method supporting cross-platform service invocation in cloud environment was studied [9]. The authors put forward a Local Optimization and Enumeration Method (LOEM) for finding a QoS near-to-optimal composite and a

decision-making method for supporting cross-platform service invocation in cloud environment. Graph-based service composition was also employed for the purpose of decreasing the complexity of the service composition task [10]. Tao et al. developed a parallel intelligent algorithm named full connection based parallel adaptive chaos optimization with reflex migration (FC-PACO-RM) for RSC optimal selection in CMfg [11]. Recently an increasing number of heuristic searching methods and swarm-based intelligent algorithms are employed to solve the problem, such as the genetic algorithm (GA) [8]. Wu and Zhu introduced an approach that combines the transaction-aware service composition, the QoS-aware service composition, and the ant colony optimization (ACO) together to achieve a trade-off between accuracy and time-efficiency of solving RSC [12]. This algorithm is also exploited for web service composition [13]:

$$q'(s_i) = \begin{cases} \dfrac{q(s_i) - q_{\min}(s_i)}{q_{\max}(s_i) - q_{\min}(s_i)}, & \text{for positive properties} \\ \dfrac{q_{\max}(s_i) - q(s_i)}{q_{\max}(s_i) - q_{\min}(s_i)}, & \text{for negative properties.} \end{cases} \quad (1)$$

In addition to the aforementioned RSC methods concerning the objective QoS properties, trust-QoS as a subjective property has also received extensive attention for the reasons presented in previous paragraphs. It is a subjective degree of belief about specific agents [14]. It has been used in proliferation in conducting electronic commerce or business of the Internet in service-oriented environments. In service-oriented architecture, trust is normally divided into four categories: direct experience, trusted third parties, hybrid, and trust negotiation, each of them was described in [15] but the authors did not provide the calculation method. The evaluation models of trust-QoS in MG were proposed [16], in which the trust falls into intradomain and interdomain. The authors also gave a trust-QoS-based MG resource service scheduling framework based on this trust aggregation method. Hammadi et al. [17] used probability theory to determine the trustworthiness of the composite service in the sequential and parallel composite structures. However, probability theory is usually used to determine objective uncertainties and is therefore inappropriate for measuring subjective trust degree. Fuzzy logic and fuzzy number were said to be suitable for modeling the trust in peer-to-peer (P2P) environment, or the trust between users and each business services [18, 19]. The above researches provide the basis of the trust and QoS properties aggregation methods to our model and will not be reiterated in detail.

2.2. The Basic Bees Algorithm. The basic Bees Algorithm is a member of the swarm-based algorithm community. It is inspired by foraging behavior of honeybees in nature [20]. A colony of honeybees can extend itself over a long distance and in multiple directions to exploit the most profitable food sources for survival of its colony. During the selection of its nest sites and food resources, they have shown particular characteristics such as precisely navigated, wisely decision-making loop, and well organized. The basic Bees Algorithm is featured by the combination of local and global search, which correlate with the process of exploitation and exploration, respectively. It starts by sending n scout bees in the solution space for sampling. These scouts are sorted according to their sampled fitness evaluated by objective functions. The algorithm then selects m scouts with better fitness for local search (or neighborhood search) in the neighbor of the scouts. The neighborhood size is specified by *ngh*. Out of the m selected scouts, the top ranked e (less than m) scouts recruit *nre* followers while the rest of the selected scouts recruit *nrb* (less than *nre* due to the lower fitness) followers. During this process, two strategies are employed to enhance the performance [21]. (1) The neighborhood size shrinks if a cycle of local search fails to yield better fitness. (2) A stagnated solution is abandoned to help the algorithm escape from local optima, and the stagnation limit is specified by stlim. Meanwhile, the scouts unqualified for neighborhood search scatter in the solution space randomly again. This step intends to explore unidentified regions that may potentially contain prominent solutions and to maintain the diversity of the scout population. Additionally, predefined stopping criteria are needed to decide when the algorithm should terminate. The Bees Algorithm has been used with great success to calculate almost optimal solutions to a large number of problems like function optimizing and various engineering problems, such as cell formation [22], mechanical design [23], printed-circuit board assembly optimization [24], control systems tuning [25], filter design [26], pattern recognition [27], and chemical engineering [28]. In this paper, the basic Bees Algorithm will be made suitable for solving RSC optimization problem in a multiuser manufacturing environment. Detailed presentation will be provided in the following two sections.

3. Formation of the Multiuser RSC Model

A multiuser RSC model is established in this section. Several assumptions should be observed for the establishment. (1) All the manufacturing task requests launched by different users are homogeneous, meaning the requested tasks can be broken down into the same series of subtasks and each subtask shares the candidate service pool individually. (2) The consideration of QoS properties is by no mean exhaustive in this model. Only three QoS properties are exemplified, namely, time, cost, and trust, since the principles of evaluating different aggregated QoS properties have been studied intensively and are analogous in the same composite architecture. (3) All encapsulated stand-alone resource services will not be distinguished from whether they are from the elastic cloud platform or the internet through the outsourcing process.

3.1. Denotations. Provided the number of task requests posted at a time is I, the request from the user U_i is represented by

$$\text{Task}_i = \{\text{st}_1, \text{st}_2, \ldots, \text{st}_J\} \quad (i = 1, 2, \ldots, I), \quad (2)$$

where st_j ($j = 1, 2, \ldots, J$) denotes a subtask and J the number of subtasks. The subtask st_j can be functionally finished by a service chosen from the service pool:

$$\mathrm{Pool}_j = \{\mathrm{as}_1, \mathrm{as}_2, \ldots, \mathrm{as}_M\}, \tag{3}$$

where as_m ($m = 1, 2, \ldots, M$) represents a candidate stand-alone service (or atom service) in Pool_j and M specifies the number of stand-alone services that Pool_j contains.

A task request can be responded functionally by a service chain consisting of the selected stand-alone services from each service pool. A service chain can be denoted by

$$\mathrm{Chain}_i = \left\{\mathrm{as}^1, \mathrm{as}^2, \ldots, \mathrm{as}^J\right\}, \tag{4}$$

where as^j ($j = 1, 2, \ldots, J$) symbolizes one of the services chosen from Pool_j.

Let R_i be the task request by U_i, and it can be expressed as a three-tuple:

$$R_i = \left\langle \mathrm{Task}_i^r, \mathrm{time}_i^r, \mathrm{cost}_i^r \right\rangle, \tag{5}$$

where Task_i^r denotes the user's functional description of the task, while time_i^r and cost_i^r stand for the user's QoS requirements for the manufacturing time and cost, respectively. In the model, the users' QoS requirements are scaled into the interval $(0, 1)$ and categorized into four degrees that are low: $(0.5, 1.0)$, mediate: $(0.25, 0.75)$, high: $(0.15, 0.65)$, and very high: $(0, 0.5)$.

Let Q_i be the QoS evaluation of a composite service, and it can be also represented as a three-tuple:

$$Q_i = \left\langle \mathrm{trust}_i, \mathrm{time}_i, \mathrm{cost}_i \right\rangle, \tag{6}$$

where trust_i, time_i, and cost_i denote the aggregated trust, time, and cost evaluations of a composite service, respectively.

3.2. Trust Management. In this model, trust is managed in two perspectives: (1) direct trust denoted by $\mathrm{dTrust}_i^{(k)}(\mathrm{as}^j)$, where k counts the number of transactions with the service as^j. It only concerns with those services that the user has directly collaborated with; (2) indirect trust or recommended trust denoted by $\mathrm{inTrust}_i^{(k)}(\mathrm{as}^j)$, which is a continuation of direct trust and is more concerned with other users' feedback of direct experience. The method of quantifying the two types of trust is not detailed as there are too many related researches and beyond the scope of this paper. The aggregated trust is calculated as

$$\mathrm{Trust}_i^k\left(\mathrm{as}^j\right) = \begin{cases} \alpha \cdot \mathrm{dTrust}_i^k\left(\mathrm{as}^j\right) + (1-\alpha) \cdot \mathrm{inTrust}_i^k\left(\mathrm{as}^j\right), & \mathrm{dTrust}_i^k\left(\mathrm{as}^j\right) \neq 0 \\ \mathrm{inTrust}_i^k\left(\mathrm{as}^j\right), & \mathrm{dTrust}_i^k\left(\mathrm{as}^j\right) = 0, \end{cases} \tag{7}$$

where α signifies the user's inclination to depend more on self-experience, so α satisfies $0.5 < \alpha < 1$.

A trusted service set (TSS) is introduced to each user, which is expressed as

$$\mathrm{TSS}_i = \{\mathrm{as}_l \mid l \leq L\}, \tag{8}$$

where as_l symbolizes a trusted services belonging to the user U_i and L symbolizes the maximum trusted services a user is allowed to have. A stand-alone service is enrolled into TSS only if the number of satisfactory transaction with it in a certain period exceeds a predefined threshold. The next time when the user publishes a task request, the services in the TSS take a priority of being chosen. The TSS is made to be a sequence. It is updated dynamically by the sequence tail (the earliest enrolled trusted service) being discarded and the latest qualified one being inserted into the sequence head. This technique attempts to imitate people's social behavior that they are more likely to socialize with those who they are recently made acquainted and trusted.

3.3. Objectives of the Model. Three service composite modes are considered in the model, namely, one-to-one mapping (O2O), many-to-one mapping (M2O), and many-to-many mapping (M2M). Each service chain fulfills only one and at least one task in the O2O mode. This mode is applied to when there are plenty of resource services available. The mode of M2O fits into the condition that users post tough time requirements, so several service chains are allowed to complete one task in cooperation. And the M2M mode is applicable when the resource services become very scarce and it is barely possible to allocate at least one service chain to every task separately. In this mode a resource service can be shared among different users to tradeoff the resource scarcity.

The objectives of the model are established as follows:

$$\max \quad \sum_{i=1}^{I} \mathrm{Trust}_i^{(k)} \tag{9}$$

$$\max \quad \sum_{i=1}^{I} x_i \tag{10}$$

$$\begin{aligned} \text{s.t.} \quad & \mathrm{time}_i < \mathrm{time}_i^r, \quad i = 1, 2, \ldots, I \\ & \mathrm{cost}_i < \mathrm{cost}_i^r, \quad i = 1, 2, \ldots, I. \end{aligned} \tag{11}$$

The objective (9) realizes the maximization of the aggregated trust that all users hold to the composite service. In the objective (10), x_i is a decision variable. It equals 1 if the task posted by U_i is completed successfully whereas it equals 0 if unsuccessfully. This objective requires the model to satisfy as more users as possible. As to the constraints (11), they point out that the manufacturing time and cost of should be in accordance with users' demands individually. Only when both of them answer to the user's specifications the transaction can be regarded as successful. As can be seen, the established model considers all the users' requirements as a whole to realize a more reasonable resource allocation.

4. The Bees Algorithm for the Problem

This section introduces the modified Bees Algorithm as a combinational multiobjective constraint-handling optimizer for the established multiuser RSC model.

4.1. Solution Schema. In the modified Bees Algorithm, each scout honeybee represents a solution (a composite service)

that answers the calls of users. In the composition mode of O2O each scout can be expressed as

$$S = \begin{bmatrix} \left\{as_1^1, as_1^2, \ldots, as_1^J\right\} \\ \left\{as_2^1, as_2^2, \ldots, as_2^J\right\} \\ \vdots \\ \left\{as_I^1, as_I^2, \ldots, as_I^J\right\} \end{bmatrix} \quad (12)$$

in which the number of service chain is I equals the number of users. The stand-alone services composing the solution will be referred as component service in the algorithm. The format of scout in the M2O and M2M modes can be deduced similarly and are not given here.

4.2. Scouts Initialization. The initialization process starts by examining whether the user's TSS is empty. If not, a service will be selected randomly from the TSS for building a service chain; otherwise a service will be selected on a random basis from the service pool. In the modes of O2O and M2O, a stand-alone service is not allowed to be shared by several different services chains, meaning an invoked service will not be able to answer the call of other users. However, in the M2M mode, service chain sharing is permitted to compensate for the resource scarcity. Therefore the number of times that a service can be invoked is not restricted. Totally, the number of scout bees to be initialized is n.

4.3. Waggle Dance. The n scout bees are ranked into Pareto dominated sets after being evaluated by objective function and constraints. The purpose is to find Pareto optimality. In Pareto optimality solutions cannot be improved in any of the objectives without degrading at least one of the other objectives. Particular rules are designed as a constraint-handling strategy to sort out the dominated solution sets. In the rules, a feasible solution refers to the one that satisfies all the constraints, whereas an infeasible solution means that it violates at least one constraint. The rules are presented as follows.

(1) Any feasible solution shall dominate any infeasible solution.

(2) Feasible solutions with better objective fitness shall dominate.

(3) For two feasible solutions that have identical objective fitness, the one with larger value for positive properties (or smaller value for negative properties) shall dominate.

(4) Among two infeasible solutions, the one violates less constraints shall dominate.

(5) Among two infeasible solutions that have identical constraint violations, the one with larger value for positive properties (or smaller value for negative properties) being violated shall dominate.

(6) In other cases, the solutions are categorized into the same Pareto set.

After obtaining the Pareto dominated sets, all scout bees are sorted into a sequence according to their dominance relationships. Scouts in the same set are sorted on a random basis. Then, nre foraging bees are recruited by the top e scout bees and nrb foraging bees by the next $m - e$ scout bees for neighborhood search. The rest of $n - m$ scout bees are not qualified for recruiting foraging bees and will thus perform random global search.

4.4. Neighborhood Search. Probability-based neighborhood search and neighborhood shrinking strategies are developed to facilitate the search in discrete domain. The initial neighborhood size ngh is set to be a probability p_0. Each component service in a scout bee correlates with a random number uniformly generated in the interval $(0, 1)$. Each foraging bee searches around the scout bee in the form that the component services are replaced by other services in the pool if their corresponding numbers are smaller than p_0, while the others remain as components as before. Afterwards, the dominance between the foraging bee and the scout bee is determined using the rules introduced in Section 4.3. If a foraging bee can dominate the scout bee, it will replace the scout bee and become the new scout in the next search iteration. In the algorithm, this is said improvement has been produced. Otherwise the scout bee retains and the neighborhood size shrinks according to $ngh = \gamma \cdot ngh$, where γ is the shrinking coefficient. The failure of producing improvement after a certain number of iterations will force the scout bees abandon its position and reallocate a random one. This step demonstrates the site abandonment strategy in the Bees Algorithm and used when the scout bee is believed to have been trapped in local optima.

4.5. Global Search. The remaining $n - m$ scout bees not qualified for neighborhood search perform the random global search in the solution space. These scout bees are randomized again as they are in the initialization step. This is intended to maintain the diversity of the scout population and explore the solution space to discover potential excellent solutions.

4.6. Flowchart and Time Complexity Analysis. The flowchart of the Bees Algorithm for finding an optimal solution to the model is presented by Figure 1. The modified Bees Algorithm uses the parameter nomenclatures stated in previous paragraphs, and the time complexity in one algorithm iteration can be then analyzed as follows:

(1) scout bees initialization: $T_1 = O(n \times I \times J)$,

(2) waggle dance: $T_2 = O(n^2)$,

(3) fitness evaluation: $T_3 = O(n \times I \times J)$,

(4) neighborhood search: $T_4 = O(e \times nre \times I \times J + (m - e) \times nrb \times I \times J)$,

(5) global search: $T_5 = O((n - m) \times I \times J)$,

of which (2)–(5) are in the main loop. The low power components can be negligible with the scale of the problem

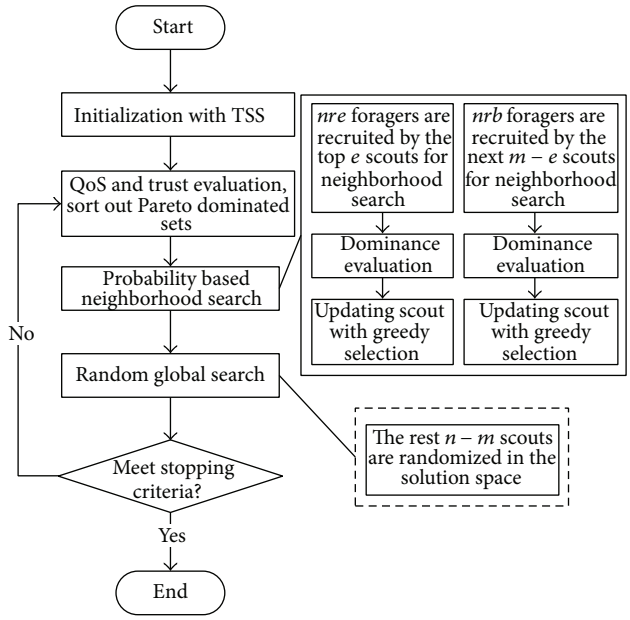


Figure 1: Flowchart of the Bees Algorithm.

Table 2: Parameter settings for the RSC model.

Mode	Group one		Group two		Group three
	O2O	M2O	O2O	M2O	M2M
Subtask number	7	7	10	10	7
Service number	15	15	25	25	4
User number	5	5	5	5	7
Request number	3	3	3	3	5

increases, provided the number of iteration to is K, and the time complexity of the modified Bees Algorithm is

$$T = O\left([(e \times nre) + (m - e) \times nre + (2n - m)] \times I \times J \times K\right). \quad (13)$$

5. Experiments and Discussions

5.1. Experimental Setups. Experiments are implemented in three main groups of different complexity under different RSC modes, as shown in Table 2. Groups one and two demonstrate how the composition modes O2O and M2O deal with problems of different complexity. Group three illustrates the way that the mode M2M deals with the situation of insufficient services. For all the experiments, the transaction cycles are set to be 75, and the values of all stand-alone service QoS properties are generated randomly. Moreover, all the users' time and cost requirements fall into their individual range of "low," "mediate," "high," and "very high" on a random basis. For simplification, only the sizes of solution space for O2O in group one and two are calculated, which are $15^7 \times 14^7 \times 13^7 \approx 1.13 \times 10^{24}$ and $25^7 \times 24^7 \times 23^7 \approx 9.53 \times 10^{28}$, respectively. The modified Bees Algorithm uses the parameter settings as given in Table 3 for different experiment groups unless being specified. Success rate is used as a critical metric to evaluate the effectiveness of the algorithm. It is the number of users whose task requests have been satisfied divided by the total number of users who have published their task requests.

Table 3: Parameter settings for the Bees Algorithm.

	Group one	Group two	Group three
ns	10	16	10
nb	4	4	4
ne	1	1	1
nrb	5	5	5
nre	10	10	10
stlim	7	7	7
ngh	0.6	0.6	0.6

5.2. Success Rate. The established model introduces the TSS to every user as described previously. Its effectiveness is investigated in the modes of O2O and M2O. The performance of the algorithm using FSS is compared with the algorithm without FSS, as shown in Figure 2. It can be seen that the success rate drops as the users' time and cost requirements become tough simultaneously; meanwhile, the algorithm using TSS outperforms the one without it. This pattern is true for both group one and two. To be specific, the algorithm performs well in dealing with low and mediate requirements (both time and cost), as more than 90% users' requests being satisfied in the mode O2O of group one. Particularly, it reaches 100% success rate when the users have merely low requirements, regardless of the use of USS. However, the success rate drops to around 60% and 38% when the users' requirements become high and very high, respectively. Nevertheless, the algorithm using TSS is able to promote the success rate by approximately 4% on average. The average promotions of success rate are 7.3%, 11.4%, and 11.2% in Figures 2(b), 2(c), and 2(d), respectively. Therefore the utilization of TSS for each user is significant for the algorithm to search for a near optimal composite service at high success rate.

5.3. Avoiding Fraud Services. The model is capable of preventing user from invoking fraud services to a degree by setting the users' trust as one of the objective functions. The fraud service in the experiment is defined as the service that does not provide genuine QoS information when registering as an encapsulated member of the service pool. It has lower QoS values than it claims but is still capable of completing the task functionally. Both the users' time and cost requirements are kept as mediate throughout this test. The proportions of fraud service in the service pool are set to increase from 20% to 80% with the step of 20%. Details of the result can be referred to Figure 3. In Figures 3(a) and 3(b), which are obtained from group one by the modes O2O and M2O, respectively, the curves imply that an increasing number of fraud services are invoked with the growth of the fraud service proportion in the service pool, and the success rate declines due to this growth. For the algorithm without TSS, only 1.9%, 7.0% and 1.9%, 3.8% of the fraud services have

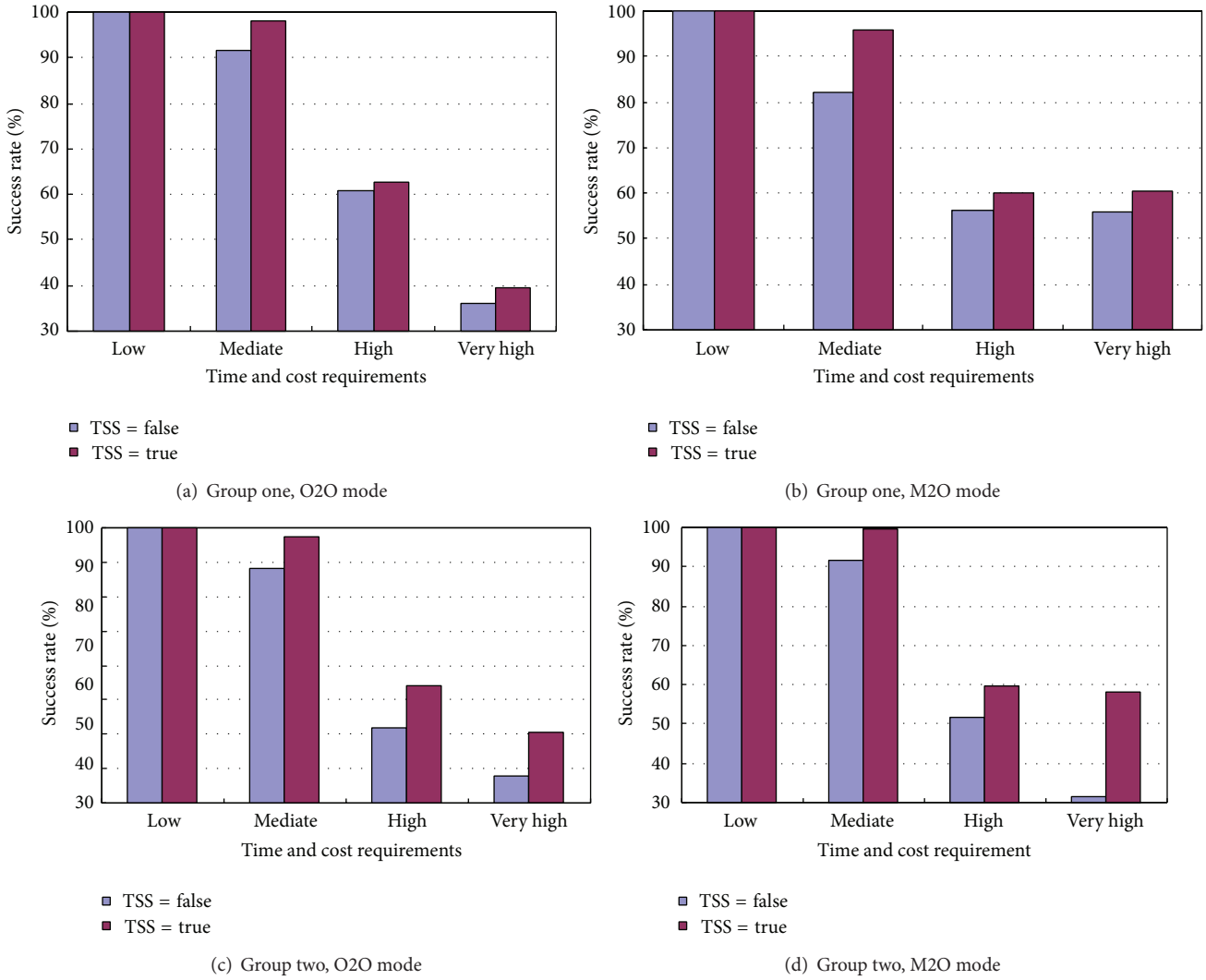
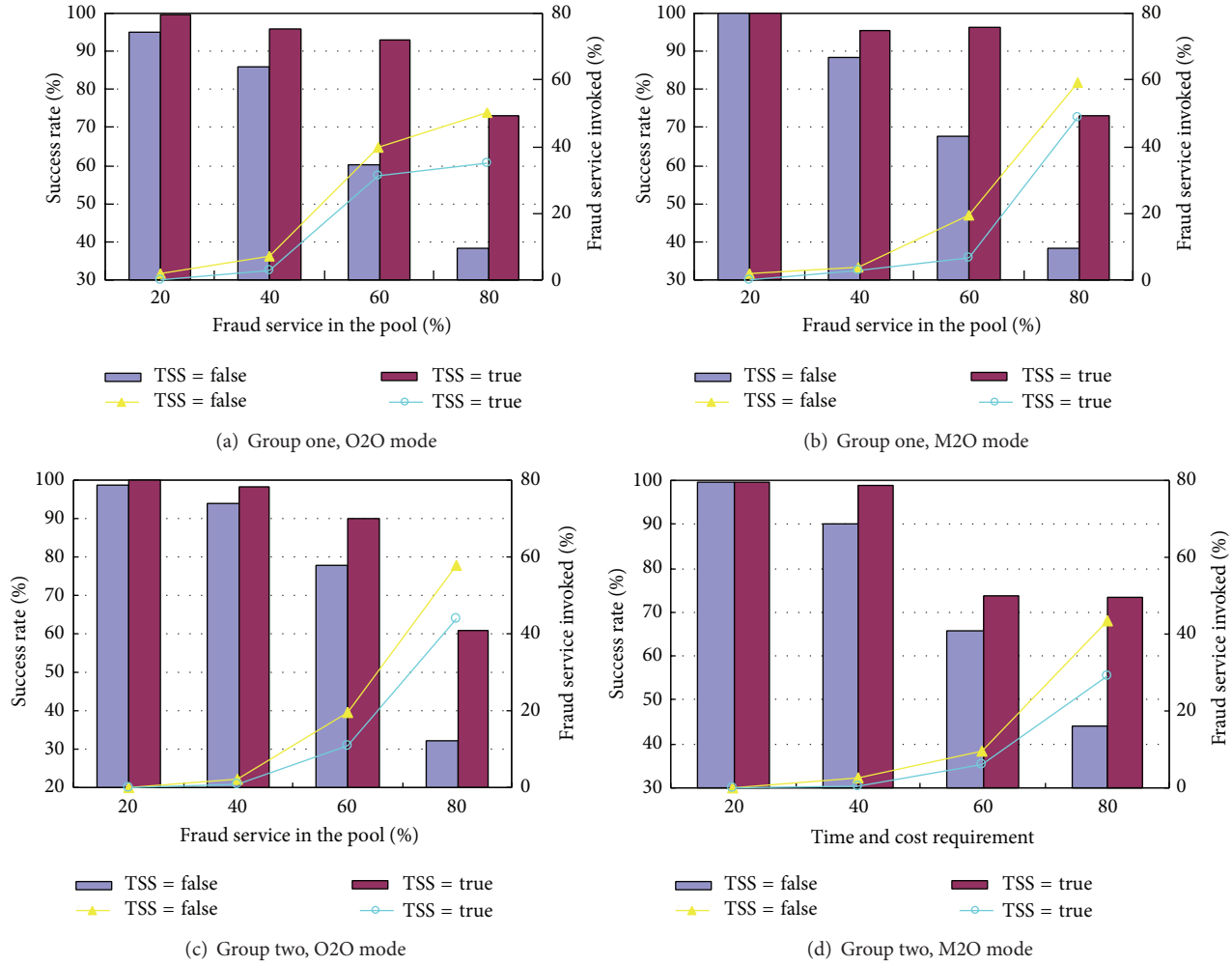


Figure 2: Comparison between the success rates with and without TSS.

been integrated into the composite service by O2O and M2O, respectively, when there are 20% and 40% fraud services in the service pool, with 95.1%, 85.8% and 100%, 88.4% tasks being successfully fulfilled. As the proportion of the fraud service climbs up to 60% and 80%, 39.6%, 50.0%, and 19.4%, 59.4% fraud services are invoked, respectively, by O2O and M2O, with the corresponding success rates further down to 60.4%, 38.2% and 67.6%, 38.2%. Figures 3(c) and 3(d) display the results from group two, where a similar trend can be seen. However, two vital points can be summarized from the figures that underline the advantages of the use of trust aggregation and TSS: (1) although the invoked fraud service increases with the growth of fraud service percentage in the service pool, the ratio of them being invoked in the composite service is much lower than its actual ratio in the service pool. For example, in the experiment of group two under the O2O mode, the composite service found by the algorithm without TSS contains 0.1%, 2.3%, 19.4%, and 57.8% fraud services when there are actually 20%, 40%, 60%, and 80% fraud service in the pool. It can be calculated that approximately 30% fraud services are avoided on average. (2) The ratio of fraud services included in the composite service can be further reduced if TSS is considered. Again in group two under the O2O mode, the ratio of fraud service being invoked is decreased to 0.1%, 0.9%, 10.8%, and 44.1%, which is further 5.9% lower on average than the algorithm without TSS. Consequently the success rate is raised from 98.7%, 93.8%, 78.0%, and 32.0% to 100%, 98.2%, 90.2%, and 60.9%, respectively, with the growth of fraud service in the pool from 20% to 80%. Hence this experiment indicates that the evaluation of users' aggregated trust can help the users avoid ingenuity to a substantial degree. In fact the use of TSS plays a significant role to further enhance the ability to avoid ingenuity and thus promote the composite success rate.

5.4. Composite Speed. The purpose of this experiment is to investigate the impact of TSS on the speed of searching a near optimal composite service. Also, both the users' time and cost requirements are kept the same in the four degrees. The results obtained from group one and two in the modes



(a) Group one, O2O mode

(b) Group one, M2O mode

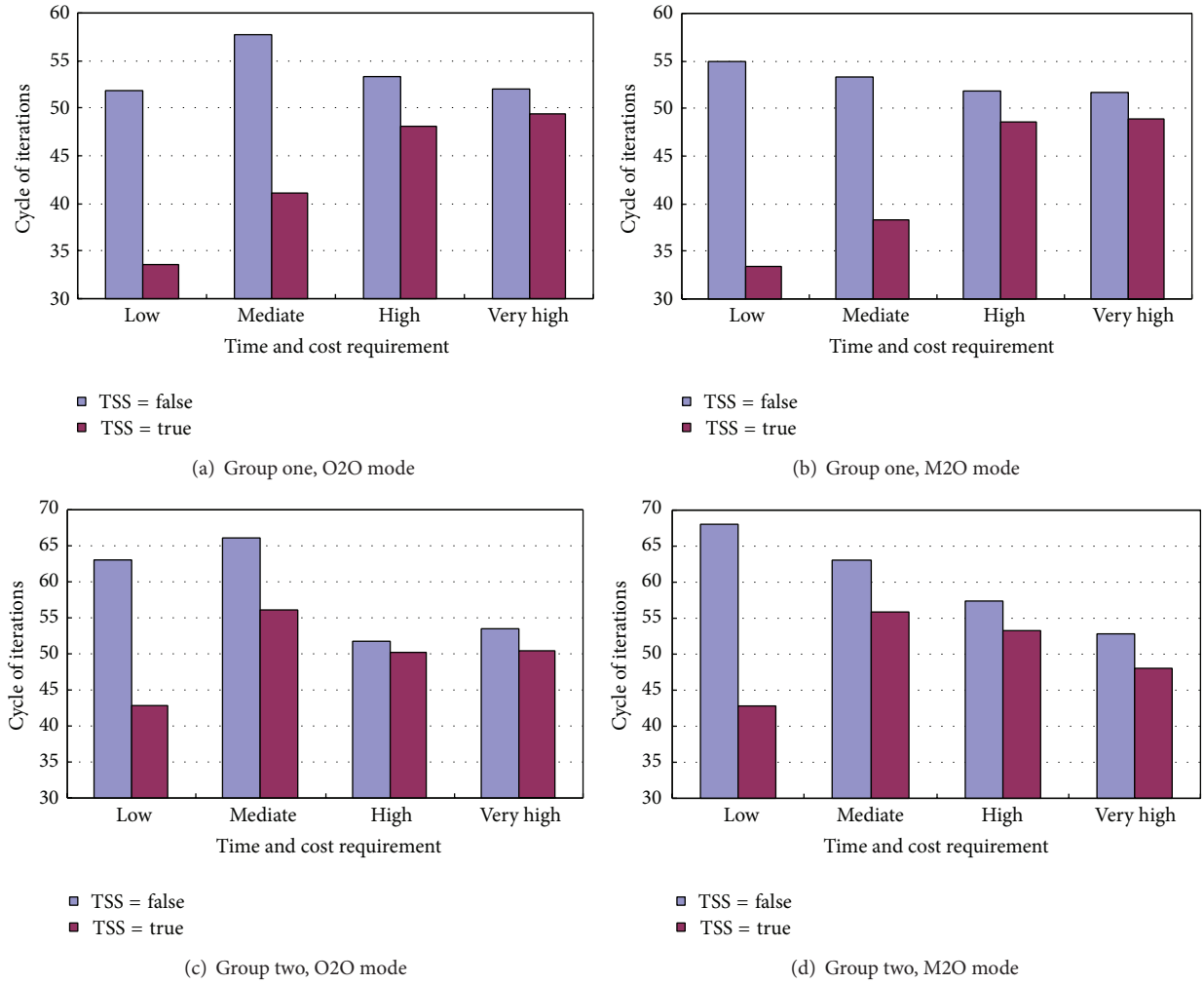
(c) Group two, O2O mode

(d) Group two, M2O mode

Figure 3: Ability of dealing with fraud services.

O2O and M2O are shown in Figure 4. All the four bar graphs, which represent four varying experimental settings, are characterized by one feature that the bars on the right are lower than those on the left. The feature indicates that the introduction of TSS assists the algorithm in finding an optimal composite faster. The experimental data tells that the TSS helps reduce 18.3, 21.5, 20.2, and 25.2 iteration cycles in dealing with users' low requirements, and 16.7, 14.9, 10.0, and 7.14 in dealing with mediate requirements. The low and mediate requirements are the two situations of which the algorithm can finish the tasks at high success rate. However, it can be observed that the algorithm using TSS does not holds clear advantage over its counterpart when facing the users' high and very high time and cost requirements. As the users' requirements become stringent, the unsuccessful service composition increases. The iteration cycles for an unsuccessful service composition do not make practical sense for statistical calculation; therefore the bars representing the iteration cycles for high and very high requirements do not provide typical characteristic of the speed of successful searches.

5.5. Ability of M2O to Handle Various Time Requirements. The mode M2O is particularly introduced to deal with demanding time requirement. It allows several service chains to work in collaboration for performing one task when it is not possible for one service chain or very hard to find one to complete the required task. This experiment is intended to address the advantage of M2O over O2O in this particular situation. In the experiment, the cost requirement is kept constant as mediate while the time requirement grows from low to very high. Figure 5 gives the comparison between the performances of the two modes in group one and two. It is understandable that the performances of the two modes in handling low and mediate time requirements do not vary too much, and the use of TSS further reduces this variance. However, the advantage of the M2O mode becomes apparent when the time requirement keeps growing. Take the results presented in Figure 5(b) to demonstrate, if the users only have low or mediate time requirements, the M2O mode fulfills the tasks with 0.5% and 0.7% higher success rate than the O2O mode. However, the M2O mode produces 23.4% and 13.3%



(a) Group one, O2O mode

(b) Group one, M2O mode

(c) Group two, O2O mode

(d) Group two, M2O mode

Figure 4: Comparison between the composite speed with and without TSS.

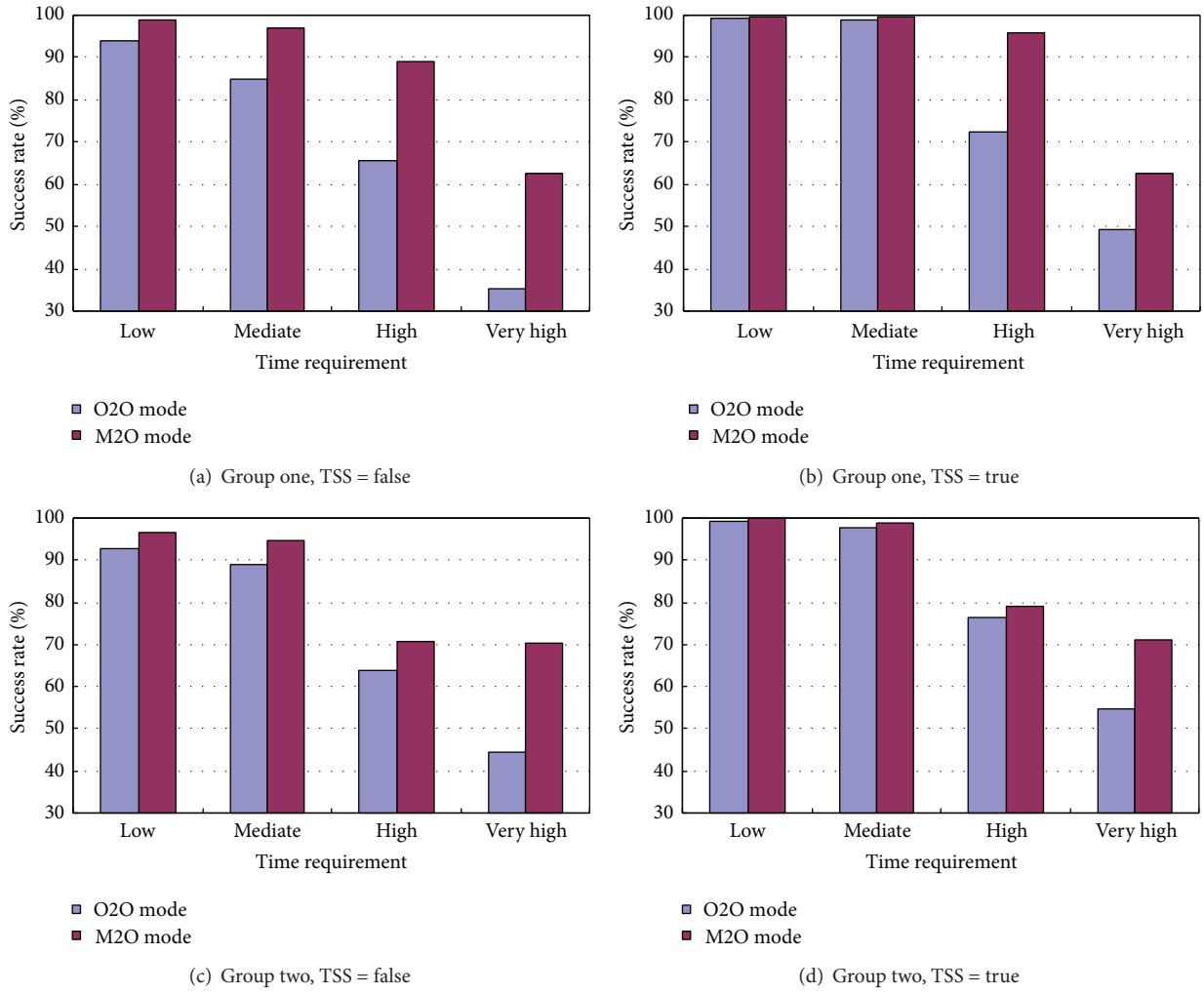
Table 4: Parameter settings for different colony sizes.

	Groups one and three						Group two				
	c1	c2	c3	c4	c5	c6	c1	c2	c3	c4	c5
ns	3	5	10	20	30	40	4	10	16	28	40
nb	1	2	4	8	12	16	1	4	4	7	10
ne	1	1	1	2	3	4	1	1	1	1	2
nrb	5	5	5	5	5	5	5	5	5	5	5
nre	10	10	10	10	10	10	10	10	10	10	10
Colony size	13	20	35	70	105	140	14	35	41	68	100

higher success rate than the O2O modes in handling high and very high time requirements, respectively, due to the time-saving collaborative working mechanism.

5.6. Impact of Colony Size to the Success Rate. Various parameter combinations are used with the aim of investigating how the bee colony size can influence the composite results. Related parameters are listed in Table 4. Some unmentioned parameters like stlim remain their previous values. Both the users' time and cost requirements are kept unchanged to be mediate throughout this experiment.

It can be calculated that the colony size grows from the combination c1 to c6 in group one, and c1 to c5 in group two. In this experiment, group one and two do not share a common parameter setting because the RSC problems in the two groups do not have the same complexity. The impact of the colony size on the composite success rate is demonstrated in Figure 6. The curves in the figures imply the tendency of the success rate as the colony size grows. Generally, it can be seen in the four graphs that the increase of colony size enhances the composite success rate. Specifically, noticeable enhancement of the success rate can be observed if the colony size grows from a very small size. However, the enhancement of success rate is not very noticeable if the colony size continues to grow. It is analyzed that the algorithm is able to find a near optimal composite service when its colony grow to certain size as large as c3 or c4 in this experiment. Further augment of the colony will not bring considerable benefit any more. In Figure 6(a), for instance, without using the TSS, the composite success rate climbs up from 84.9% to 87.1%, 92.9% and 97.8% as the colony size grows from 13 to 20, 35, and



(a) Group one, TSS = false

(b) Group one, TSS = true

(c) Group two, TSS = false

(d) Group two, TSS = true

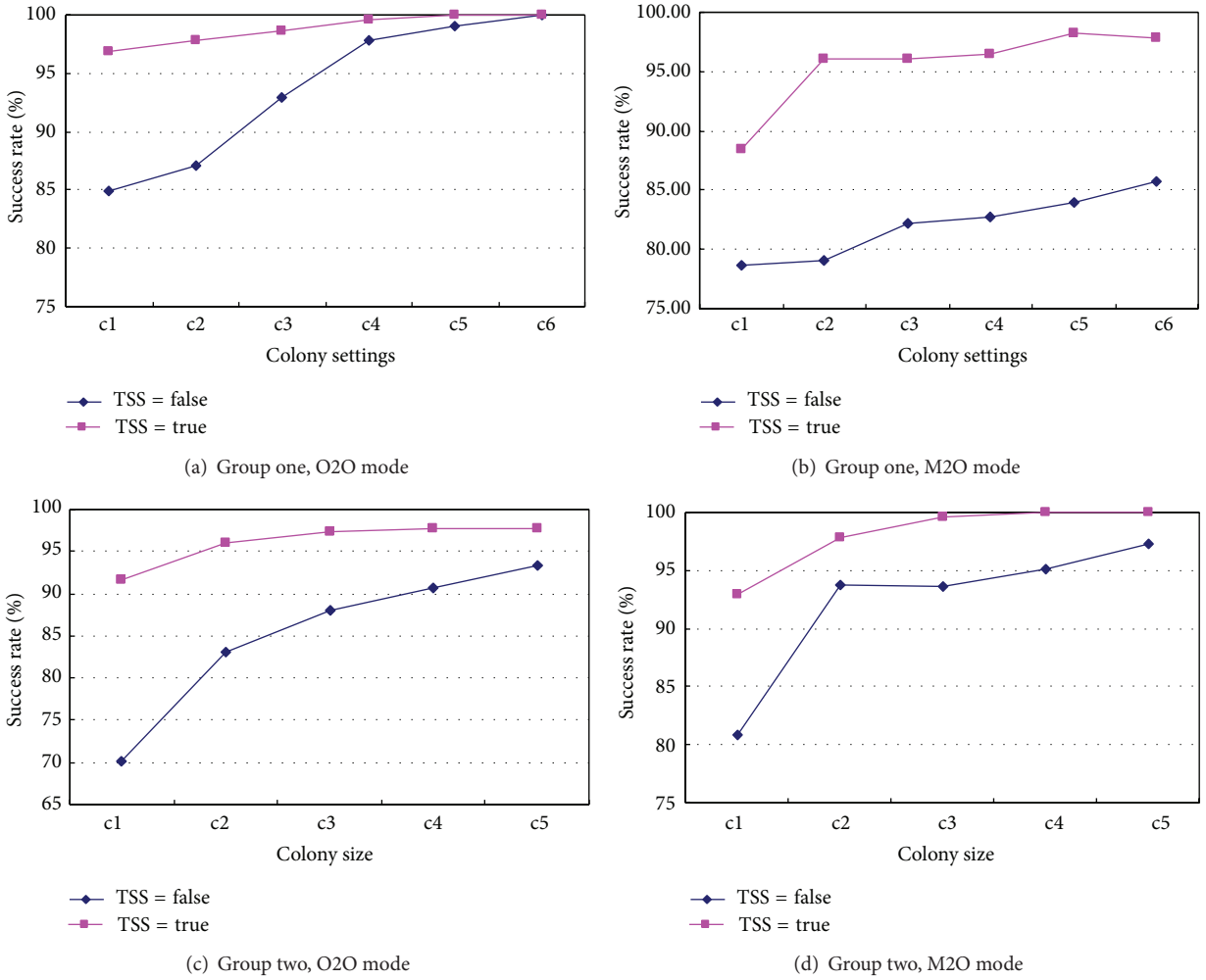
Figure 5: Performance of M2O in handling different time requirements.

70, respectively. The promoted percentage is 2.2%, 5.8%, and 4.9%. When the colony size continues to increase to 105 and 140, the corresponding success rate promotion is down to 1.3% and 0.9%. This pattern can match that in Figures 6(b), 6(c), and 6(d).

5.7. Investigation on the M2M Mode. The M2M mode is applicable to where many users post task requests but barely sufficient resource services are available. Due to the service insufficiency, the TSS and fraud services are not considered in this experiment in the hope of exploring the potential of all services. Only the effectiveness of the algorithm and the impact of its colony size are investigated. The results are given in Figure 7. It can be seen that the composite success rate can still reach 100% when the users have low time and cost requirements, and this rate remains as high as 89.0% when the users raise their requirements to mediate. The rate continues to decrease to 64.8% and 16.5% as the users post high and very high requirements. A decreasing number of users can be satisfied as they raise their requirement. But the effectiveness of the M2M mode cannot be overlooked because it performs well in solving the problem of insufficient resource service when dealing with low and mediate requirements. In addition, Figure 7(b) tells that the growth of the colony size contributes to the promotion of the composite success rate. However, the enhancement does not endure if the colony size exceeds a certain scale, which resembles the trend in the O2O and M2O modes in the preceding experiments.

6. Conclusion

Traditionally, resource service composition methods respond to merely one task request at a time, which could easily lead to misallocation of resources. These methods always search for a composition service of the best quality for one user but ignore a comprehensive QoS trade-off among multiple users. A multiuser RSC method for solving the above problem has been presented and tested on the basis of the established multiuser RSC model. In this model, trust is considered as one of the objectives and QoS requirements as constraints. Particularly, the TSS as a new trust-aware technique is utilized and the experimental results have validated its effectiveness



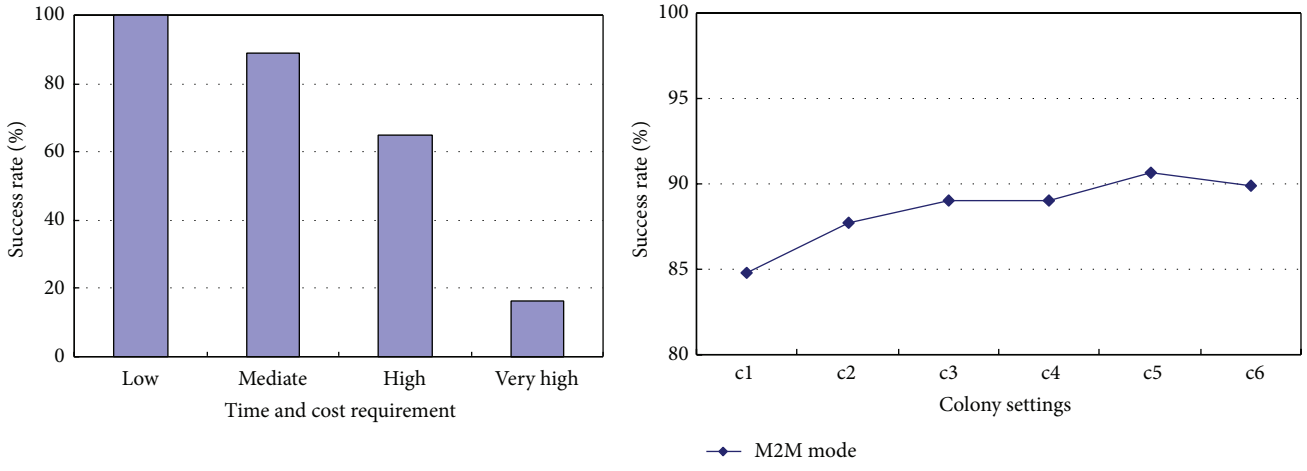
(a) Group one, O2O mode

(b) Group one, M2O mode

(c) Group two, O2O mode

(d) Group two, M2O mode

Figure 6: Impact of the colony size on the success rate.



(a) Group three, effectiveness test

(b) Group three, colony size test

Figure 7: Results obtained in the M2M mode.

in terms of the enhancement of success rate, the ability to avoid service, and the speed to find a feasible solution. Furthermore, the advantage of the M2O mode in handling tough time requirement and the M2M mode in dealing with resource service insufficiency is also reflected by the results.

The research has not yet been fully completed. The model in this paper does not consider heterogeneous and dynamic task requests. The fault tolerance ability of the model is not investigated either. Furthermore, the investigation on the M2M mode needs to be extended to a more generic study. These problems remain as future research topics.

Conflict of Interests

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

Acknowledgments

This research is supported by the International Science & Technology Cooperation Program of China (Grant no. 2015DFA70340), the National Natural Science Foundation of China (Grant no. 51175389), the Key Project of Natural Science Foundation of Hubei Province of China (Grant no. 2013CFA044), and the Research Exchange with China and India, The Royal Academy of Engineering (Grant no. 1415-1).

References

[1] F. Tao, L. Zhang, V. C. Venkatesh, Y. Luo, and Y. Cheng, "Cloud manufacturing: a computing and service-oriented manufacturing model," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 225, no. 10, pp. 1969–1976, 2011.

[2] L. Z. Zeng, B. Benatallah, A. H. H. Ngu, and M. Dumas, "QoS-ware middleware for web service composition," *IEEE Transactions on Software Engineering*, vol. 30, no. 5, pp. 311–327, 2004.

[3] G. Canfora, M. D. Penta, R. Esposito, and M. L. Villani, "A framework for QoS-aware binding and re-binding of composite web services," *Journal of Systems and Software*, vol. 81, no. 10, pp. 1754–1769, 2008.

[4] S. X. Sun and J. Zhao, "A decomposition-based approach for service composition with global QoS guarantees," *Information Sciences*, vol. 199, pp. 138–153, 2012.

[5] W. Zhou, J. H. Wen, M. Gao, and J. W. Liu, "A QoS preference-based algorithm for service composition in service-oriented network," *Optik*, vol. 124, no. 20, pp. 4439–4444, 2013.

[6] M. Alrifai and T. Risse, "Combining global optimization with local selection for efficient QoS-aware service composition," in *Proceedings of the 18th International World Wide Web Conference (WWW '09)*, pp. 881–890, ACM, Madrid, Spain, April 2009.

[7] L. Y. Qi, Y. Tang, W. C. Dou, and J. J. Chen, "Combining local optimization and enumeration for QoS-aware web service composition," in *Proceedings of the IEEE International Conference on Web Services*, pp. 34–41, July 2010.

[8] F. Mardukhi, N. NematBakhsh, K. Zamanifar, and A. Barati, "QoS decomposition for service composition using genetic algorithm," *Applied Soft Computing Journal*, vol. 13, no. 7, pp. 3409–3421, 2013.

[9] L. Qi, W. C. Dou, X. Y. Zhang, and J. J. Chen, "A QoS-aware composition method supporting cross-platform service invocation in cloud environment," *Journal of Computer and System Sciences*, vol. 78, no. 5, pp. 1316–1329, 2012.

[10] J. Rao, P. Küngas, and M. Matskin, "Composition of semantic web services using linear logic theorem proving," *Information Systems*, vol. 31, no. 4-5, pp. 340–360, 2006.

[11] F. Tao, Y. Laili, L. Xu, and L. Zhang, "FC-PACO-RM: a parallel method for service composition optimal-selection in cloud manufacturing system," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 4, pp. 2023–2033, 2013.

[12] Q. W. Wu and Q. S. Zhu, "Transactional and QoS-aware dynamic service composition based on ant colony optimization," *Future Generation Computer Systems*, vol. 29, no. 5, pp. 1112–1119, 2013.

[13] S. R. Dhore and M. U. Kharat, "QoS based web services composition using ant colony optimization: mobile agent approach," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 1, no. 7, pp. 519–527, 2012.

[14] B. A. Misztal, *Trust in Modern Societies*, Polity Press, Cambridge, UK, 1996.

[15] H. D. Field, Q. Shi, M. Merabti, and R. Askwith, "Trust in service composition," in *Proceedings of the 11th Annual Conference on the Convergence of Telecommunications, Networking & Broadcasting (PGNet '10)*, Liverpool , UK, January 2010.

[16] F. Tao, Y. F. Hu, D. Zhao, and Z. D. Zhou, "An approach to manufacturing grid resource service scheduling based on trust-QoS," *International Journal of Computer Integrated Manufacturing*, vol. 22, no. 2, pp. 100–111, 2009.

[17] A. M. Hammadi, T. S. Dillon, and E. Chang, "Business service composability on the basis of trust," in *Proceedings of the 3rd IEEE International Conference on Digital Ecosystems and Technologies (DEST' 09)*, pp. 437–440, Istanbul, Turkey, June 2009.

[18] N. Griffiths, K. M. Chao, and M. Younas, "Fuzzy trust for peer-to-peer based systems," in *Proceedings of the 26th IEEE International Conference on Distributed Computing Systems Workshops (ICDCS 06)*, pp. 73–79, July 2006.

[19] S. Li, Y. Fan, and X. Li, "A trust-based approach to selection of business services," *International Journal of Computer Integrated Manufacturing*, vol. 24, no. 8, pp. 769–784, 2011.

[20] D. T. Pham, A. Ghanbarzadeh, E. Koc, S. Otri, S. Rahim, and M. Zaidi, "The bees algorithm—a novel tool for complex optimisation problems," in *Proceedings of the 2nd International Virtual Conference on Intelligent Production Machines and Systems (IPROMS '06)*, pp. 454–459, Elseviers, Oxford, UK, 2006.

[21] D. T. Pham and M. Castellani, "The bees algorithm: modelling foraging behaviour to solve continuous optimization problems," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 223, no. 12, pp. 2919–2938, 2009.

[22] D. T. Pham, A. A. Afify, and E. Koc, "Manufacturing cell formation using the bees algorithm," in *Proceedings of the 3rd International Virtual Conference on Intelligent Production Machines and Systems (IPROKMS '07)*, Whittles, Dunbeath, UK, 2007.

[23] D. T. Pham, M. Castellani, and A. Ghanbarzadeh, "Prelininary design using the bees algorithm," in *Proceedings of the 8th LAMDAMAP International Conference on Laser Metrology, CMM and Machine Tool Performance*, pp. 420–429, Cardiff, UK, 2007.

[24] M. C. Ang and D. T. Pham, "Application of the bees algorithm with TRIZ-inspired operation for PCB assembly planning," in

Proceedings of the 5th Virtual International Conference on Innovative Production Machines and Systems, pp. 405–410, Cardiff, UK, 2009.

[25] D. Pham, A. H. Darwish, and E. Eldukhri, "Optimisation of a fuzzy logic controller using the Bees Algorithm," *International Journal of Computer Aided Engineering and Technology*, vol. 1, no. 2, pp. 250–264, 2009.

[26] D. T. Pham and E. Koç, "Design of a two-dimensional recursive filter using the Bees Algorithm," *International Journal of Automation and Computing*, vol. 7, no. 3, pp. 399–402, 2010.

[27] D. T. Pham and A. Haj Darwish, "Using the bees algorithm with Kalman filtering to train an artificial neural network for pattern classification," *Proceedings of the Institution of Mechanical Engineers. Part I: Journal of Systems and Control Engineering*, vol. 224, no. 7, pp. 885–892, 2010.

[28] M. Castellani, Q. T. Pham, and D. T. Pham, "Dynamic optimisation by a modified bees algorithm," *Proceedings of the Institution of Mechanical Engineers Part I: Journal of Systems and Control Engineering*, vol. 226, no. 7, pp. 956–971, 2012.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 587923, 10 pages
http://dx.doi.org/10.1155/2015/587923

Research Article

Incremental Discriminant Analysis in Tensor Space

Liu Chang,[1,2] Zhao Weidong,[1,2] Yan Tao,[1,2] Pu Qiang,[1,2] and Du Xiaodan[1,2]

[1]*College of Information Science and Technology, Chengdu University, Chengdu 610106, China*
[2]*Key Laboratory of Pattern Recognition and Intelligent Information Processing in Sichuan, Chengdu 610106, China*

Correspondence should be addressed to Du Xiaodan; duxiaodandan@cdu.edu.cn

Received 20 September 2014; Revised 4 January 2015; Accepted 8 February 2015

Academic Editor: Rahib H. Abiyev

To study incremental machine learning in tensor space, this paper proposes incremental tensor discriminant analysis. The algorithm employs tensor representation to carry on discriminant analysis and combine incremental learning to alleviate the computational cost. This paper proves that the algorithm can be unified into the graph framework theoretically and analyzes the time and space complexity in detail. The experiments on facial image detection have shown that the algorithm not only achieves sound performance compared with other algorithms, but also reduces the computational issues apparently.

1. Introduction

Nowadays, increasing amounts of data in the field of industrial, economic, medical, and other application areas, such as signals, measurements, images, and videos, are becoming available due to the development of computer technology. In order to excavate the hidden information in the data implicitly describing underlying processes or structures, advanced intelligent tools are proposed. However, since the stochastic nature of the processes and their measurement, structure in this data is mostly collected with noise. Consequently, it is reasonable to seek robust and adaptive tools that can cope with this nature.

Computational intelligence techniques have been investigated to answer this need. These techniques have been concerned with reproducing the abilities of human brains. Machine learning techniques exactly imitate the learning procedure of human, which construct learning model based on example data and use that to make predictions and decisions. However, due to the noise in data, it is important to construct efficient learning model to help sift useful information from the noise.

In regards to this, machine learning algorithms project high-dimensional data into low-dimensional feature space to make their low-features as separable as possible. Generally, they are classified into two categories: supervised learning and unsupervised learning. The essential difference between supervised learning and unsupervised learning is that whether the class information is considered. Generally speaking, the recognition performance of supervised learning is superior to that of unsupervised learning. As a classical machine learning algorithm, linear discriminate analysis (LDA) [1, 2] seeks optimal discriminative vectors to maximize the interclass scatter matrix and to minimize the intraclass scatter matrix. A large number of research works have shown the predominant advantage of LDA in various applications.

It is worth noting that traditional LDA is based on vector model. It requires all data being vectorized before learning. Actually, high-dimensional image data is structured data; the vectorization operation will break the correlation relationship of different pixels. Furthermore, the vectorization operation also is easy to result in the curse of dimensionality problem. As a result, machine learning algorithms [3–11] based on tensor algebra are investigated. These algorithms consider high-dimensional image as a high order tensor and introduce tensor algebra to analyze tensor data. Tensor representation not only is helpful to preserve the structure of high-dimensional image, but also serves as an effective way to avoid the curse of dimensionality problem. To unify all machine learning algorithms, [12] proposes the graph embedding framework. Under this graph embedding framework, two kinds of projective forms are summarized, called vector-to-vector and tensor-to-tensor forms, respectively.

However, for all machine learning algorithms, they have to train all samples again when new samples are added, which results in heavy computational cost. Consequently, incremental machine learning algorithms are proposed [13–17]. But most incremental learning algorithms focus on vector machine learning. Only a limited number of works study incremental learning in tensor space [18–20]. To investigate the incremental tensor learning, this paper develops incremental tensor discriminant analysis (ITDA), which employs supervised learning in tensor space and introduces incremental learning to process online learning. Furthermore, as a kind of machine learning algorithm, this paper also exploits the relationship between the proposed methods and the graph embedding framework and proves that the algorithm is a special case of tensor-to-tensor form under the graph embedding framework theoretically. This paper also analyzes the time and space complexity in detail. At last, this paper conducts facial image detection experiments to evaluate the proposed method. The experimental results have demonstrated the advantage of the method.

2. Tensor Discriminant Analysis

For multidimensional image data $X = \{X_1, \ldots, X_K\}$, where $X_i \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, the corresponding class label is $l(i) \in [1, C]$, where C is the number of the class. Let the number of the cth class be n_c; then the following definitions are introduced.

Definition 1. Within-class scatter tensor is defined:

$$S_w = \sum_{c=1}^{C} \sum_{i=1}^{n_c} \left\| X_i - \overline{X}_c \right\|^2, \tag{1}$$

where $\overline{X}_c$ represent the mean tensor of the cth class.

Definition 2. Between-class scatter tensor is defined:

$$S_b = \sum_{c=1}^{C} n_c \left\| \overline{X}_c - \overline{X} \right\|^2, \tag{2}$$

where $\overline{X}$ represent the total mean tensor.

Definition 3. Total scatter tensor is defined:

$$S_t = \sum_{i=1}^{K} \left\| X_i - \overline{X} \right\|^2. \tag{3}$$

It is easy to derive that

$$\begin{aligned} S_w + S_b &= \sum_{c=1}^{C} \sum_{i=1}^{n_c} \left\| X_i - \overline{X}_c \right\|^2 \\ &\quad + \sum_{c=1}^{C} n_c \left\| \overline{X}_c - \overline{X} \right\|^2 \\ &= \sum_{c=1}^{C} \sum_{i=1}^{n_c} \left\| \mathrm{vec}\left(X_i\right) - \mathrm{vec}\left(\overline{X}_c\right) \right\|^2 \\ &\quad + \sum_{c=1}^{C} n_c \left\| \mathrm{vec}\left(\overline{X}_c\right) - \mathrm{vec}\left(\overline{X}\right) \right\|^2 \\ &= \sum_{i=1}^{K} \left\| \mathrm{vec}\left(X_i\right) - \mathrm{vec}\left(\overline{X}\right) \right\|^2 \\ &= S_t. \end{aligned} \tag{4}$$

Definition 4. Mode-n within-class scatter matrix is defined:

$$S_w^{(n)} = \sum_{c=1}^{C} \sum_{i=1}^{n_c} \left(X_i^{(n)} - \overline{X}_c^{(n)} \right) \left(X_i^{(n)} - \overline{X}_c^{(n)} \right)^T, \tag{5}$$

where $X_i^{(n)}$ is the mode-n matrix of the ith sample and $\overline{X}_c^{(n)}$ is the mode-n mean matrix of the cth class.

Definition 5. Mode-n between-class scatter matrix is defined:

$$S_b^{(n)} = \sum_{c=1}^{C} n_c \left(\overline{X}_c^{(n)} - \overline{X}^{(n)} \right) \left(\overline{X}_c^{(n)} - \overline{X}^{(n)} \right)^T, \tag{6}$$

where $\overline{X}^{(n)}$ is the mode-n total mean matrix.

Definition 6. Mode-n total scatter matrix is defined:

$$\begin{aligned} S_t^{(n)} &= \sum_{i=1}^{K} \left(X_i^{(n)} - \overline{X}^{(n)} \right) \left(X_i^{(n)} - \overline{X}^{(n)} \right)^T \\ &= S_b^{(n)} + S_w^{(n)}. \end{aligned} \tag{7}$$

The basic idea of TDA is to seek N projective matrices to make within-class scatter tensor smaller and between-class scatter tensor larger. The objective function is

$$\begin{aligned} &J\left(U^{(1)}, \ldots, U^{(N)}\right) \\ &= \arg\max \frac{\sum_{c=1}^{C} n_c \left\| \overline{Y}_c - \overline{Y} \right\|^2}{\sum_{c=1}^{C} \sum_{i=1}^{n_c} \left\| Y_i - \overline{Y}_c \right\|^2} \\ &= \arg\max \frac{\sum_{c=1}^{C} n_c \left\| \left(\overline{X}_c - \overline{X} \right) \times {}_n U^{(n)^T} \Big|_{n=1}^{N} \right\|^2}{\sum_{c=1}^{C} \sum_{i=1}^{n_c} \left\| \left(X_i - \overline{X}_c \right) \times {}_n U^{(n)^T} \Big|_{n=1}^{N} \right\|^2}. \end{aligned} \tag{8}$$

In order to solve the above function, the iterative technique is adopted. It is assumed that the projective matrices

$\{U^{(1)}, \ldots, U^{(n-1)}, U^{(n+1)}, \ldots, U^{(N)}\}$ are known; then $U^{(n)}$ is solved as follows:

$$\begin{aligned} J\left(U^{(n)}\right) &= \arg\max\left(\left(\sum_{c=1}^{C} n_c U^{(n)^T}\left(\overline{X}_c^{(n)} - \overline{X}^{(n)}\right)U^{(-n)}U^{(-n)^T}\right.\right. \\ &\quad \cdot \left.\left(\overline{X}_c^{(n)} - \overline{X}^{(n)}\right)^T U^{(n)}\right) \\ &\quad \cdot \left(\sum_{c=1}^{C}\sum_{i=1}^{n_c} U^{(n)^T}\left(X_i^{(n)} - \overline{X}_c^{(n)}\right)U^{(-n)}U^{(-n)^T}\right. \\ &\quad \cdot \left.\left.\left(X_i^{(n)} - \overline{X}_c^{(n)}\right)^T U^{(n)}\right)^{-1}\right), \end{aligned} \tag{9}$$

where $U^{(-n)} = U^{(N)} \cdots \otimes U^{(n+1)} \otimes U^{(n-1)} \cdots \otimes U^{(1)}$. Since $U^{(-n)}U^{(-n)^T} = I$, so the above equation can be rewritten:

$$\begin{aligned} J\left(U^{(n)}\right) &= \arg\max \frac{\sum_{c=1}^{C} n_c U^{(n)^T}\left(\overline{X}_c^{(n)} - \overline{X}^{(n)}\right)\left(\overline{X}_c^{(n)} - \overline{X}^{(n)}\right)^T U^{(n)}}{\sum_{c=1}^{C}\sum_{i=1}^{n_c} U^{(n)^T}\left(X_i^{(n)} - \overline{X}_c^{(n)}\right)\left(X_i^{(n)} - \overline{X}_c^{(n)}\right)^T U^{(n)}} \\ &= \arg\max \frac{U^{(n)^T} S_b^{(n)} U^{(n)}}{U^{(n)^T} S_w^{(n)} U^{(n)}}. \end{aligned} \tag{10}$$

Based on the basic concept of TDA and related matrix knowledge, we can get the following theorems.

Theorem 7. *In tensor discriminant analysis, the mode-n intraclass scatter matrix is generally nonsingularity.*

Proof. Defining the following matrix

$$H_w^{(n)} = \left[X_1^{(n)} - \overline{X}_{l(1)}^{(n)}, \ldots, X_M^{(n)} - \overline{X}_{l(M)}^{(n)}\right], \tag{11}$$

where M is the number of samples, $l(m)$ expresses the class label of the mth sample. Then the mode-n intraclass scatter matrix is represented:

$$S_w^{(n)} = H_w^{(n)} H_w^{(n)^T}, \tag{12}$$

where $H_w^{(n)} \in \mathbb{R}^{I_n \times D}$, $D = \prod_{k=1,k\neq n}^{N} I_k$. Generally speaking, $I_n \ll D$; then

$$\operatorname{rank}\left(H_w^{(n)}\right) = \min\left(I_n, D\right) = I_n. \tag{13}$$

Further, we can get

$$\operatorname{rank}\left(S_w^{(n)}\right) = \operatorname{rank}\left(H_w^{(n)} H_w^{(n)^T}\right) = \operatorname{rank}\left(H_w^{(n)}\right) = I_n. \tag{14}$$

So the theorem is proved. □

Theorem 8. *Equation (8) can be unified into the graph embedding framework [12].*

Proof. Based on the basic concept of tensor algebra, the numerator of (8) can be rewritten:

$$\begin{aligned} &\sum_{c=1}^{C}\sum_{i=1}^{n_c}\left\|Y_i - \overline{Y}_c\right\|^2 \\ &\quad = \sum_{c=1}^{C}\sum_{i=1}^{n_c}\left\|\operatorname{vec}\left(Y_i\right) - \operatorname{vec}\left(\overline{Y}_c\right)\right\|^2. \end{aligned} \tag{15}$$

Letting $y_i = \operatorname{vec}(Y_i)\operatorname{vec}(\overline{Y}_c) = (1/n_c)\sum_{j=1}^{n_c} y_j$, then the above equation is written:

$$\begin{aligned} &\sum_{c=1}^{C}\sum_{i=1}^{n_c}\left\|y_i - \frac{1}{n_c}\sum_{j=1}^{n_c} y_j\right\|^2 \\ &\quad = \sum_{c=1}^{C}\sum_{i=1}^{n_c}\left(y_i - \frac{1}{n_c}\sum_{j=1}^{n_c} y_j\right)\left(y_i - \frac{1}{n_c}\sum_{j=1}^{n_c} y_j\right)^T \\ &\quad = \sum_{i=1}^{K} y_i y_i^T - \sum_{c=1}^{C}\frac{1}{n_c}\sum_{l(i)=l(j)=c} y_i y_j^T \\ &\quad = \sum_{i=1}^{K}\sum_{j=1}^{K} W_{ij}^{(w)} y_i y_i^T - \sum_{i,j=1}^{K} W_{ij}^{(w)} y_i y_j^T \\ &\quad = \frac{1}{2}\sum_{i=1}^{K}\sum_{j=1}^{K} W_{ij}^{(w)} y_i y_i^T + \frac{1}{2}\sum_{i=1}^{K}\sum_{j=1}^{K} W_{ij}^{(w)} y_i y_i^T \\ &\qquad - \frac{1}{2}\sum_{i,j=1}^{K} W_{ij}^{(w)} y_i y_j^T \\ &\qquad - \frac{1}{2}\sum_{i,j=1}^{K} W_{ij}^{(w)} y_i y_j^T \\ &\quad = \frac{1}{2}\sum_{i,j=1}^{K} W_{ij}^{(w)}\left(y_i y_i^T + y_j y_j^T - y_i y_j^T - y_j y_i^T\right) \\ &\quad = \frac{1}{2}\sum_{i,j=1}^{K} W_{ij}^{(w)}\left(y_i - y_j\right)\left(y_i - y_j\right)^T \\ &\quad = \frac{1}{2}\sum_{i,j=1}^{K} W_{ij}^{(w)}\left\|Y_i - Y_j\right\|^2, \end{aligned} \tag{16}$$

where

$$W_{ij}^{(w)} = \begin{cases} \dfrac{1}{n_c}, & l(i) = l(j) = c, \\ 0, & l(i) \neq l(j). \end{cases} \tag{17}$$

Within the low-dimensional feature space, it is desired to preserve the property as demonstrated in (4), so the denominator of (8) is formulated as follows:

$$\begin{aligned}
&\sum_{c=1}^{C} n_c \left\| \overline{Y}_c - \overline{Y} \right\|^2 \\
&= \sum_{c=1}^{C} n_c \left\| \mathrm{vec}\left(\overline{Y}_c\right) - \mathrm{vec}\left(\overline{Y}\right) \right\|^2 \\
&= \sum_{i=1}^{K} \left\| \mathrm{vec}\left(\overline{Y}_i\right) - \mathrm{vec}\left(\overline{Y}\right) \right\|^2 \\
&\quad - \sum_{c=1}^{C} \sum_{i=1}^{n_c} \left\| \mathrm{vec}\left(Y_i\right) - \mathrm{vec}\left(\overline{Y}_c\right) \right\|^2 \\
&= \sum_{i=1}^{K} \left(y_i - \overline{y}\right)\left(y_i - \overline{y}\right)^T \\
&\quad - \sum_{c=1}^{C} \sum_{i=1}^{n_c} \left\| \mathrm{vec}\left(Y_i\right) - \mathrm{vec}\left(\overline{Y}_c\right) \right\|^2,
\end{aligned} \tag{18}$$

where

$$\begin{aligned}
&\sum_{i=1}^{K} \left(y_i - \overline{y}\right)\left(y_i - \overline{y}\right)^T \\
&= \sum_{i=1}^{K} y_i y_i^T - \frac{1}{K} \sum_{i,j=1}^{K} y_i y_j^T \\
&= \sum_{i=1}^{K} \left(\sum_{j=1}^{K} \frac{1}{K} \right) y_i y_i^T - \frac{1}{K} \sum_{i,j=1}^{K} y_i y_j^T \\
&= \frac{1}{2} \left(\frac{1}{K} \sum_{i,j=1}^{K} y_i y_i^T + \frac{1}{K} \sum_{i,j=1}^{K} y_i y_i^T - \frac{1}{K} \sum_{i,j=1}^{K} y_i y_j^T - \frac{1}{K} \sum_{i,j=1}^{K} y_i y_j^T \right) \\
&= \frac{1}{2} \sum_{i,j=1}^{K} \frac{1}{K} \left(y_i y_i^T + y_j y_j^T - y_i y_j^T - y_j y_i^T \right).
\end{aligned} \tag{19}$$

Combining (19) with (16), (18) can be written:

$$\begin{aligned}
&\sum_{c=1}^{C} n_c \left\| \overline{Y}_c - \overline{Y} \right\|^2 \\
&= \frac{1}{2} \sum_{i,j=1}^{K} \left(\frac{1}{K} - W_{ij}^{(w)} \right) \left(y_i y_i^T + y_j y_j^T - y_i y_j^T - y_j y_i^T \right) \\
&= \frac{1}{2} \sum_{i,j=1}^{K} W_{ij}^{(b)} \left(y_i y_i^T + y_j y_j^T - y_i y_j^T - y_j y_i^T \right) \\
&= \frac{1}{2} \sum_{i,j=1}^{K} W_{ij}^{(b)} \left(y_i - y_j \right)\left(y_i - y_j \right)^T \\
&= \frac{1}{2} \sum_{i,j=1}^{K} W_{ij}^{(b)} \left\| Y_i - Y_j \right\|^2,
\end{aligned} \tag{20}$$

where

$$W_{ij}^{(b)} = \frac{1}{n} - W_{ij}^{(w)} = \begin{cases} \dfrac{1}{K} - \dfrac{1}{n_c}, & l(i) = l(j) = c, \\ \dfrac{1}{K}, & l(i) \neq l(j). \end{cases} \tag{21}$$

Consequently, (8) is expressed:

$$\begin{aligned}
&J\left(U^{(1)}, \ldots, U^{(N)}\right) \\
&= \arg\max \frac{\sum_{i,j=1}^{K} W_{ij}^{(b)} \left\| Y_i - Y_j \right\|^2}{\sum_{i,j=1}^{K} W_{ij}^{(w)} \left\| Y_i - Y_j \right\|^2} \\
&= \arg\max \frac{\sum_{i,j=1}^{K} W_{ij}^{(b)} \left\| \left(X_i - X_j\right) \times {}_n U^{(n)^T} \Big|_{n=1}^{N} \right\|^2}{\sum_{i,j=1}^{K} W_{ij}^{(w)} \left\| \left(X_i - X_j\right) \times {}_n U^{(n)^T} \Big|_{n=1}^{N} \right\|^2}.
\end{aligned} \tag{22}$$

The form of (22) is consistent with the tensor-to-tensor form of the graph embedding framework. Therefore, (8) can be unified into the graph embedding framework. □

3. Incremental Tensor Discriminant Analysis

3.1. Incremental Learning Based on a Single Sample. In order to distinguish these variables that need to be updated during incremental learning procedure, the paper employs the subscript old to mark the variables before incremental learning. For example, $\overline{X}_{\mathrm{old}}$ expresses the total mean tensor before new samples are added.

When a single sample X_{new} is added, its class label is l_{new}; then the mode-n total mean matrix becomes

$$\overline{X}^{(n)} = \frac{K \overline{X}_{\mathrm{old}}^{(n)} + X_{\mathrm{new}}^{(n)}}{K + 1}. \tag{23}$$

If $l_{\mathrm{new}} \notin [1, C]$, that is, the new sample belongs to a new class. In this case, the total class number is $C = C_{\mathrm{old}} + 1$ and mode-n interclass scatter matrix is updated:

$$\begin{aligned}
S_b^{(n)} &= \sum_{c=1}^{C} n_c \left(\overline{X}_c^{(n)} - \overline{X}^{(n)} \right)\left(\overline{X}_c^{(n)} - \overline{X}^{(n)} \right)^T \\
&= \sum_{c=1}^{C_{\mathrm{old}}} n_{\mathrm{old}_c} \left(\overline{X}_{\mathrm{old}_c}^{(n)} - \overline{X}^{(n)} \right)\left(\overline{X}_{\mathrm{old}_c}^{(n)} - \overline{X}^{(n)} \right)^T \\
&\quad + \left(X_{\mathrm{new}}^{(n)} - \overline{X}^{(n)} \right)\left(X_{\mathrm{new}}^{(n)} - \overline{X}^{(n)} \right)^T,
\end{aligned} \tag{24}$$

where n_c is the updated sample number of the cth class. Mode-n intraclass scatter matrix is

$$\begin{aligned} S_w^{(n)} &= \sum_{c=1}^{C}\sum_{i=1}^{n_c}\left(X_i^{(n)}-\overline{X}_c^{(n)}\right)\left(X_i^{(n)}-\overline{X}_c^{(n)}\right)^T \\ &= \sum_{c=1}^{C_{\text{old}}+1}\sum_{i=1}^{n_c}\left(X_i^{(n)}-\overline{X}_c^{(n)}\right)\left(X_i^{(n)}-\overline{X}_c^{(n)}\right)^T \\ &= \sum_{c=1}^{C_{\text{old}}}\sum_{i=1}^{n_{\text{old}\_c}}\left(X_i^{(n)}-\overline{X}_{\text{old}\_c}^{(n)}\right)\left(X_i^{(n)}-\overline{X}_{\text{old}\_c}^{(n)}\right)^T \\ &\quad+\left(X_{\text{new}}^{(n)}-\overline{X}_{\text{new}}^{(n)}\right)\left(X_i^{(n)}-\overline{X}_{\text{new}}^{(n)}\right)^T, \end{aligned} \tag{25}$$

where $\overline{X}_{\text{new}}^{(n)}$ is the mode-n mean matrix of the new sample. Because a single sample is added and it belongs to a new class, we can get

$$X_{\text{new}}^{(n)} = \overline{X}_{\text{new}}^{(n)}. \tag{26}$$

Then (25) becomes

$$S_w^{(n)} = \sum_{c=1}^{C_{\text{old}}}\sum_{i=1}^{n_{\text{old}\_c}}\left(X_i^{(n)}-\overline{X}_{\text{old}\_c}^{(n)}\right)\left(X_i^{(n)}-\overline{X}_{\text{old}\_c}^{(n)}\right)^T = S_{\text{old}\_w}^{(n)}. \tag{27}$$

It is demonstrated in (27) that mode-n intraclass scatter matrix will not change when a new sample with new class is added.

When the class label of the new sample $l_{\text{new}} = r \in [1, C_{\text{old}}]$, that is, the class label is not a new class. In this case, the total class number $C = C_{\text{old}}$; then mode-n interclass scatter matrix is

$$S_b^{(n)} = \sum_{c=1}^{C} n_c\left(\overline{X}_c^{(n)}-\overline{X}^{(n)}\right)\left(\overline{X}_c^{(n)}-\overline{X}^{(n)}\right)^T. \tag{28}$$

Mode-n intraclass scatter matrix is

$$\begin{aligned} S_w^{(n)} &= \sum_{c=1}^{C}\sum_{i=1}^{n_c}\left(X_i^{(n)}-\overline{X}_c^{(n)}\right)\left(X_i^{(n)}-\overline{X}_c^{(n)}\right)^T \\ &= \sum_{c=1}^{C_{\text{old}}+1}\sum_{i=1}^{n_c}\left(X_i^{(n)}-\overline{X}_c^{(n)}\right)\left(X_i^{(n)}-\overline{X}_c^{(n)}\right)^T \\ &= \sum_{\substack{c=1\\c\neq r}}^{C_{\text{old}}}\sum_{i=1}^{n_{\text{old}\_c}}\left(X_i^{(n)}-\overline{X}_{\text{old}\_c}^{(n)}\right)\left(X_i^{(n)}-\overline{X}_{\text{old}\_c}^{(n)}\right)^T \\ &\quad+\sum_{k=1}^{n_{\text{old}\_r}+1}\left(X_k^{(n)}-\overline{X}_r^{(n)}\right)\left(X_k^{(n)}-\overline{X}_r^{(n)}\right)^T. \end{aligned} \tag{29}$$

Because the new sample belongs to the rth class, then the class mean of the rth class becomes

$$\overline{X}_r^{(n)} = \frac{n_{\text{old}\_r}\overline{X}_{\text{old}\_r}^{(n)} + X_{\text{new}}^{(n)}}{n_{\text{old}\_r}+1}. \tag{30}$$

Based on this, we can get

$$\begin{aligned} &\sum_{k=1}^{n_{\text{old}\_r}+1}\left(X_k^{(n)}-\overline{X}_r^{(n)}\right)\left(X_k^{(n)}-\overline{X}_r^{(n)}\right)^T \\ &= \sum_{k=1}^{n_{\text{old}\_r}}\left(X_k^{(n)}-\overline{X}_r^{(n)}\right)\left(X_i^{(n)}-\overline{X}_r^{(n)}\right)^T \\ &\quad+\left(X_{\text{new}}^{(n)}-\overline{X}_r^{(n)}\right)\left(X_{\text{new}}^{(n)}-\overline{X}_r^{(n)}\right)^T \\ &= \sum_{k=1}^{n_{\text{old}\_r}}\left(X_k^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}+\overline{X}_{\text{old}\_r}^{(n)}-\overline{X}_r^{(n)}\right) \\ &\qquad\cdot\left(X_k^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}+\overline{X}_{\text{old}\_r}^{(n)}-\overline{X}_r^{(n)}\right)^T \\ &\quad+\left(X_{\text{new}}^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}-\frac{X_{\text{new}}^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}}{n_{\text{old}\_r}+1}\right) \\ &\qquad\cdot\left(X_{\text{new}}^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}-\frac{X_{\text{new}}^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}}{n_{\text{old}\_r}+1}\right)^T \\ &= \sum_{k=1}^{n_{\text{old}\_r}}\left(X_k^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}\right)\left(X_k^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}\right)^T \\ &\quad+\frac{n_{\text{old}\_r}}{n_{\text{old}\_r}+1}\left(X_{\text{new}}^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}\right)\left(X_{\text{new}}^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}\right)^T. \end{aligned} \tag{31}$$

So (29) is simplified:

$$\begin{aligned} S_w^{(n)} &= \sum_{\substack{c=1\\c\neq r}}^{C_{\text{old}}}\sum_{i=1}^{n_{\text{old}\_c}}\left(X_i^{(n)}-\overline{X}_{\text{old}\_c}^{(n)}\right)\left(X_i^{(n)}-\overline{X}_{\text{old}\_c}^{(n)}\right)^T \\ &\quad+\sum_{k=1}^{n_{\text{old}\_r}}\left(X_k^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}\right)\left(X_k^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}\right)^T \\ &\quad+\frac{n_{\text{old}\_r}}{n_{\text{old}\_r}+1}\left(X_{\text{new}}^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}\right)\left(X_{\text{new}}^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}\right)^T \\ &= \sum_{c=1}^{C_{\text{old}}}\sum_{i=1}^{n_{\text{old}\_c}}\left(X_i^{(n)}-\overline{X}_{\text{old}\_c}^{(n)}\right)\left(X_i^{(n)}-\overline{X}_{\text{old}\_c}^{(n)}\right)^T \\ &\quad+\frac{n_{\text{old}\_r}}{n_{\text{old}\_r}+1}\left(X_{\text{new}}^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}\right)\left(X_{\text{new}}^{(n)}-\overline{X}_{\text{old}\_r}^{(n)}\right)^T. \end{aligned} \tag{32}$$

3.2. Incremental Learning Based on Multisamples. When several samples are added, new added samples $X_{\text{new}} = \{X_{K+1}, \ldots, X_{K+T}\}$, $T \geq 1$, the corresponding class labels are $l_{\text{new}} = \{l_1, \ldots, l_T\}$. Without loss of generality, it is assumed that

$n_{\text{new\_}r}$ samples belong to the rth class; then the mean tensor of the rth class is updated:

$$\begin{aligned}\overline{X}_r &= \frac{1}{n_{\text{old\_}r}+n_{\text{new\_}r}}\left(n_{\text{old\_}r}\overline{X}_{\text{old\_}r}+\sum_{\substack{l(i)=r\\ i=1}}^{n_{\text{new\_}r}}X_i\right)\\ &= \frac{\left(n_{\text{old\_}r}\overline{X}_{\text{old\_}r}+n_{\text{new\_}r}\overline{X}_{\text{new\_}r}\right)}{n_{\text{old\_}r}+n_{\text{new\_}r}},\end{aligned} \tag{33}$$

where $\overline{X}_{\text{new\_}r}$ is the mean tensor of the new samples belonging to the rth class. The corresponding mode-n mean matrix of the rth class is

$$\overline{X}_r^{(n)} = \frac{\left(n_{\text{old\_}r}\overline{X}_{\text{old\_}r}^{n}+n_{\text{new\_}r}\overline{X}_{\text{new\_}r}^{n}\right)}{n_{\text{old\_}r}+n_{\text{new\_}r}}. \tag{34}$$

Then the number of samples in the rth is

$$n_r = n_{\text{old\_}r}+n_{\text{new\_}r}. \tag{35}$$

The total mean tensor is updated:

$$\overline{X} = \frac{K\overline{X}_{\text{old}}+\sum_{i=K+1}^{K+T}X_i}{K+T} = \frac{K\overline{X}_{\text{old}}+T\overline{X}_{\text{new}}}{K+T}, \tag{36}$$

where $\overline{X}_{\text{new}}$ is the mean tensor of all new samples. The interclass scatter mean tensor is updated:

$$S_b = \sum_{c=1}^{C}n_c\left\|\overline{X}_c-\overline{X}\right\|^2. \tag{37}$$

The corresponding mode-n interclass scatter matrix is

$$S_b^{(n)} = \sum_{c=1}^{C}n_c\left(\overline{X}_c^{(n)}-\overline{X}^{(n)}\right)\left(\overline{X}_c^{(n)}-\overline{X}^{(n)}\right)^T. \tag{38}$$

The mode-n intraclass scatter matrix is

$$\begin{aligned}S_w^{(n)} &= \sum_{c=1}^{C}\sum_{i=1}^{n_c}\left(X_i^{(n)}-\overline{X}_c^{(n)}\right)\left(X_i^{(n)}-\overline{X}_c^{(n)}\right)^T\\ &= \sum_{c=1}^{C}\left(\sum_{i=1}^{n_{\text{old\_}c}}\left(X_i^{(n)}-\overline{X}_{\text{old\_}c}^{(n)}\right)\left(X_i^{(n)}-\overline{X}_{\text{old\_}c}^{(n)}\right)^T\right.\\ &\qquad\left.+\,n_{\text{old\_}c}\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_c^{(n)}\right)\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_c^{(n)}\right)^T\right)\\ &\quad+\sum_{c=1}^{C}\left(\sum_{i=1}^{n_{\text{new\_}c}}\left(X_i^{(n)}-\overline{X}_{\text{new\_}c}^{(n)}\right)\left(X_i^{(n)}-\overline{X}_{\text{new\_}c}^{(n)}\right)^T\right.\\ &\qquad\left.+\,n_{\text{new\_}c}\left(\overline{X}_{\text{new\_}c}^{(n)}-\overline{X}_c^{(n)}\right)\left(\overline{X}_{\text{new\_}c}^{(n)}-\overline{X}_c^{(n)}\right)^T\right).\end{aligned} \tag{39}$$

Substituting (34) into the following equation, we can get

$$\begin{aligned}&n_{\text{old\_}c}\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_c^{(n)}\right)\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_c^{(n)}\right)^T\\ &\quad= n_{\text{old\_}c}\left(\overline{X}_{\text{old\_}c}^{(n)}-\frac{n_{\text{old\_}c}\overline{X}_{\text{old\_}c}^{(n)}+n_{\text{new\_}c}\overline{X}_{\text{new\_}c}^{(n)}}{n_{\text{old\_}c}+n_{\text{new\_}c}}\right)\\ &\qquad\cdot\left(\overline{X}_{\text{old\_}c}^{(n)}-\frac{n_{\text{old\_}c}\overline{X}_{\text{old\_}c}^{(n)}+n_{\text{new\_}c}\overline{X}_{\text{new\_}c}^{(n)}}{n_{\text{old\_}c}+n_{\text{new\_}c}}\right)^T\\ &\quad= \frac{n_{\text{old\_}c}n_{\text{new\_}c}^2}{\left(n_{\text{old\_}c}+n_{\text{new\_}c}\right)^2}\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_{\text{new\_}c}^{(n)}\right)\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_{\text{new\_}c}^{(n)}\right)^T\\ &\quad= \frac{n_{\text{old\_}c}n_{\text{new\_}c}^2}{n_c^2}\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_{\text{new\_}c}^{(n)}\right)\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_{\text{new\_}c}^{(n)}\right)^T.\end{aligned} \tag{40}$$

Similarly, we can get

$$\begin{aligned}&n_{\text{new\_}c}\left(\overline{X}_{\text{new\_}c}^{(n)}-\overline{X}_c^{(n)}\right)\left(\overline{X}_{\text{new\_}c}^{(n)}-\overline{X}_c^{(n)}\right)^T\\ &\quad= \frac{n_{\text{new\_}c}n_{\text{old\_}c}^2}{\left(n_{\text{old\_}c}+n_{\text{new\_}c}\right)^2}\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_{\text{new\_}c}\right)\\ &\qquad\cdot\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_{\text{new\_}c}\right)^T\\ &\quad= \frac{n_{\text{new\_}c}n_{\text{old\_}c}^2}{n_c^2}\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_{\text{new\_}c}\right)\\ &\qquad\cdot\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_{\text{new\_}c}\right)^T.\end{aligned} \tag{41}$$

Substituting (40) and (41) into (39), we can obtain

$$\begin{aligned}S_w^{(n)} &= \sum_{c=1}^{n}\left(\sum_{i=1}^{n_{\text{old\_}c}}\left(X_i^{(n)}-\overline{X}_{\text{old\_}c}^{(n)}\right)\left(X_i^{(n)}-\overline{X}_{\text{old\_}c}^{(n)}\right)^T\right.\\ &\qquad\left.+\,n_{\text{old\_}c}\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_c^{(n)}\right)\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_c^{(n)}\right)^T\right)\\ &\quad+\sum_{c=1}^{n}\left(\sum_{i=1}^{n_{\text{new\_}c}}\left(X_i^{(n)}-\overline{X}_{\text{new\_}c}^{(n)}\right)\left(X_i^{(n)}-\overline{X}_{\text{new\_}c}^{(n)}\right)^T\right.\\ &\qquad\left.+\,n_{\text{new\_}c}\left(\overline{X}_{\text{new\_}c}^{(n)}-\overline{X}_c^{(n)}\right)\left(\overline{X}_{\text{new\_}c}^{(n)}-\overline{X}_c^{(n)}\right)^T\right)\\ &= \sum_{c=1}^{n}\left(\sum_{i=1}^{n_{\text{old\_}c}}\left(X_i^{(n)}-\overline{X}_{\text{old\_}c}^{(n)}\right)\left(X_i^{(n)}-\overline{X}_{\text{old\_}c}^{(n)}\right)^T\right.\\ &\qquad\left.+\sum_{i=1}^{n_{\text{new\_}c}}\left(X_i^{(n)}-\overline{X}_{\text{new\_}c}^{(n)}\right)\left(X_i^{(n)}-\overline{X}_{\text{new\_}c}^{(n)}\right)^T\right)\end{aligned}$$

$$+\sum_{c=1}^{n}\frac{n_{\text{old\_}c}n_{\text{new\_}c}}{n_c}\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_{\text{new\_}c}^{(n)}\right)\cdot\left(\overline{X}_{\text{old\_}c}^{(n)}-\overline{X}_{\text{new\_}c}^{(n)}\right)^{T}. \tag{42}$$

Without loss of generality, it is supposed that, for T new samples, there are n_{C+1} samples belonging to the new class label $C+1$; then updated mode-n interclass scatter matrix is

$$\begin{aligned}S_b^{(n)}&=\sum_{c=1}^{C_{\text{old}}}n_c\left(\overline{X}_c^{(n)}-\overline{X}^{(n)}\right)\left(\overline{X}_c^{(n)}-\overline{X}^{(n)}\right)^{T}\\&\quad+n_{C+1}\left(\overline{X}_{C+1}^{(n)}-\overline{X}^{(n)}\right)\left(\overline{X}_{C+1}^{(n)}-\overline{X}^{(n)}\right)^{T}\\&=\sum_{c=1}^{C+1}n_c\left(\overline{X}_c^{(n)}-\overline{X}^{(n)}\right)\left(\overline{X}_c^{(n)}-\overline{X}^{(n)}\right)^{T}\end{aligned} \tag{43}$$

and mode-n intraclass scatter matrix is

$$\begin{aligned}S_w^{(n)}&=\sum_{c=1}^{C_{\text{old}}}\sum_{i=1}^{n_c}\left(X_i^{(n)}-\overline{X}_c^{(n)}\right)\left(X_i^{(n)}-\overline{X}_c^{(n)}\right)^{T}\\&\quad+\sum_{i=1}^{n_{C+1}}\left(X_i^{(n)}-\overline{X}_{C+1}^{(n)}\right)\left(X_i^{(n)}-\overline{X}_{C+1}^{(n)}\right)^{T}.\end{aligned} \tag{44}$$

It is not difficult to find that incremental learning based on singular sample only is a special case of incremental learning based on multisample.

3.3. The Complexity Analysis. For tensor discriminant analysis, the main computational time is spent on the computation of interclass mean, total mean, inter- and intraclass scatter tensor, and Eigen decomposition. The computation cost of inter- and intra-class scatter tensors depends on the number of training samples. If there are a large number of training samples, it cannot avoid to increment computational time.

For incremental discriminant analysis, the main computational time is spent on the computation of updated inter- and intraclass scatter matrix and the class number.

For Eigen decomposition, both the time complexity of TDA and ITDA are $O(NI^3)$. The main difference of the time complexity is the computation of inter- and intraclass scatter matrix. For TDA, the time complexity is $O(MNI^{N+1})$, so the time complexity will increase with the number of training samples. For ITDA, the time complexity is $O(TNI^{N+1}+CNI^{N+1})$, which is related to the class number and the number of new samples. It has no relationship with the number of initial training samples. Consequently, ITDA is helpful to reduce the time complexity.

Considering the space complexity, ITDA is also superior to TDA. When new samples are added, TDA needs $M\prod_{n=1}^{N}I_n$ bytes to save all training samples, but ITDA only needs $\prod_{n=1}^{N}I_n$ bytes to save new added samples, $\prod_{n=1}^{N}I_n$ bytes to save the total mean, $C\prod_{n=1}^{N}I_n$ bytes to save the class mean, and $N\sum_{n=1}^{N}I_n^2$ bytes to save mode-n scatter matrix. Hence ITDA has the capability to save space.

Compared to incremental learning based on single sample with incremental learning based on multisamples, incremental learning based on single samples has an advantage to reduce the space complexity because it only deals with one sample for each time.

Figure 1: The samples of CBCL dataset.

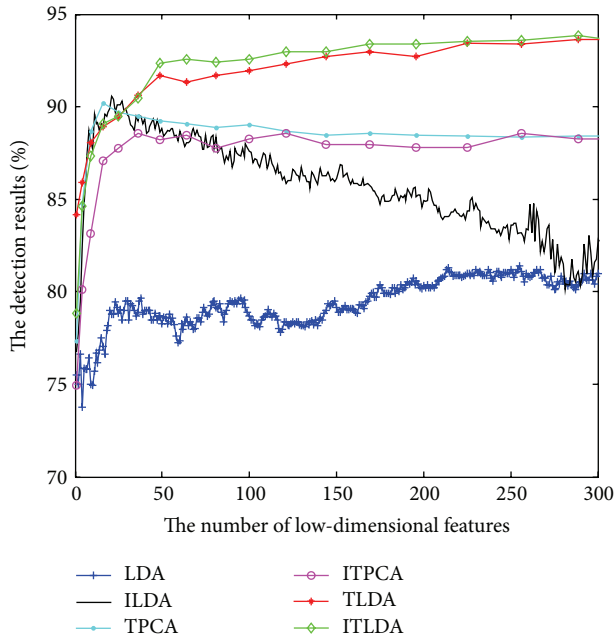


Figure 2: The detection results after the first incremental learning.

4. Experiments

In this section, a series of experiments are carried out to validate the performance of incremental tensor discriminant analysis (ITDA). The CBCL image data set is used to conduct facial image detection experiments. The dataset contains two classes of images, including facial images and nonfacial images as shown in Figure 1. The total number of the datasets is 2988 images, in which there are 2429 facial images and 559 nonfacial images. For each image, the size is 19×19. This paper divides whole dataset into training dataset with 1215 facial images and 280 nonfacial images and testing dataset with 1214 facial images and 279 nonfacial images. Furthermore, training dataset is divided into initial training

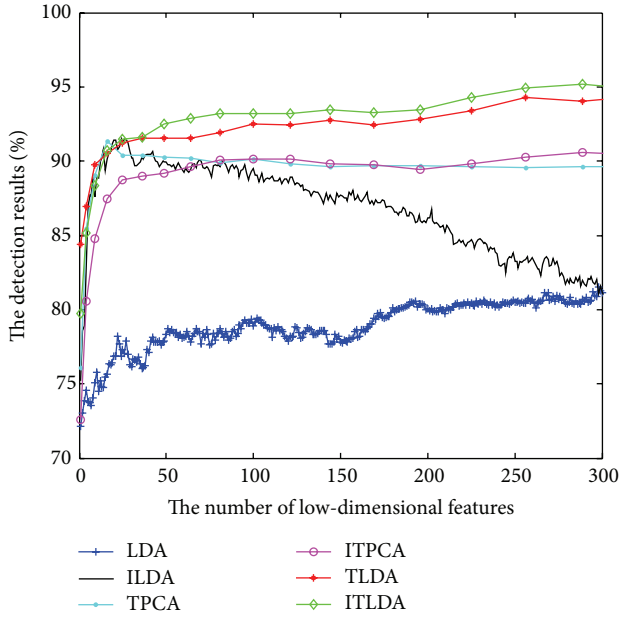


Figure 3: The detection results after the second incremental learning.

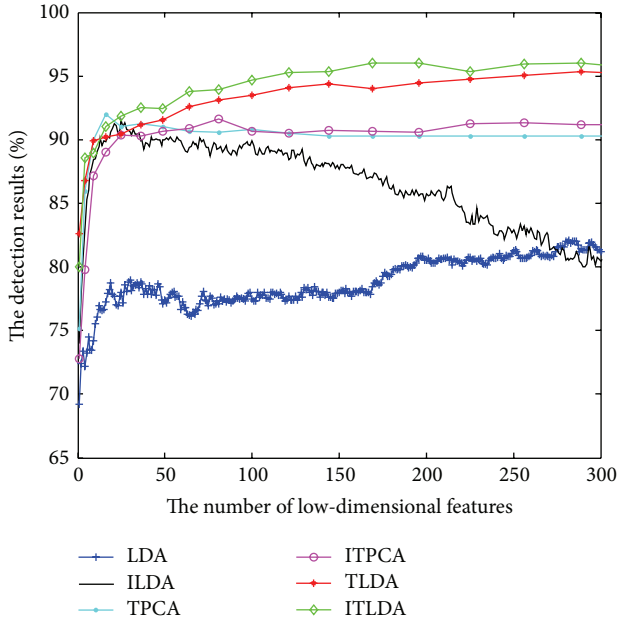


Figure 4: The detection results after the third incremental learning.

dataset with 1015 facial images and 80 nonfacial images and four incremental datasets. Each incremental dataset has 50 facial images and 50 nonfacial images.

ITLDA integrates the tensor representation and incremental learning; it is reasonable to believe that it has the advantage to improve the detection performance and reduce the time and space complexity. In this respect, ITLDA is compared with LDA [21], ILDA [14], TPCA [22], ITPCA [23], and TDA [9]. LDA is the classical linear discriminant analysis. ILDA is the incremental version of LDA. TPCA is also called MPCA (multilinear principal component analysis), which carries on principal component analysis with tensor data. ITPCA is proposed to suit for incremental principal component analysis for tensor data. TDA also represents data as tensor structure and conducts multilinear discriminant analysis. For each time of incremental learning, the paper adds one incremental dataset and then extracts low-dimensional features on testing dataset. The nearest neighbor classifier is employed to classify these low-dimensional features.

The comparisons of detection performance for different algorithms with incremental learning are shown in Figures 2, 3, 4, and 5, respectively. It is worth noting that LDA is the worst and ILDA is better than LDA. However the detection results of ILDA drop with the increment of the dimension of low-dimensional features. TPCA and ITPCA have similar detection results and both of them exceed LDA and ILDA. The probable reason is that TPCA and ITPCA represent data as tensor structure, which make full use of the interior structure information to enhance the detection performance. TDA is superior to the above four algorithms. When the dimension of low-dimensional features is low, TDA and ITDA have comparative detection percent and ITDA begins to surmount TDA when the dimension of low-dimensional features increases. Figure 6 and Table 1 have

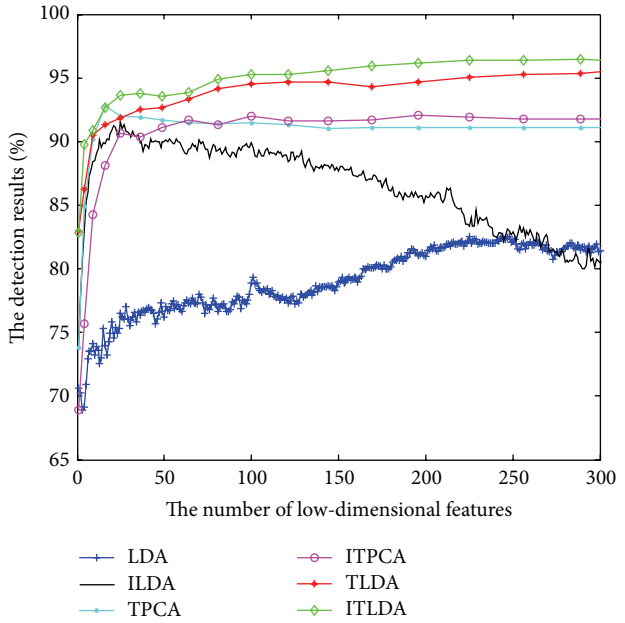


Figure 5: The detection results after the fourth incremental learning.

shown the best detection results of different algorithms. It can be seen that the detection performances of different algorithms are improved with the increment of incremental learning numbers and ITLDA always has the best performance. Consequently, it can be derived that the increment of incremental learning number is helpful to improve the detection result. More than that, as shown in Figures 7 and 8, incremental learning algorithms ILDA, ITPCA, and ITDA have the capability to alleviate time and space complexity

Table 1: The best detection results of different algorithms with incremental learning.

Algorithms	The first incremental learning (%)	The second incremental learning (%)	The third incremental learning (%)	The fourth incremental learning (%)
LDA	82.59	81.51	82.12	82.52
ILDA	90.56	91.49	91.56	91.56
TPCA	90.22	91.36	92.03	92.83
ITPCA	88.55	90.56	91.63	92.1
TLDA	93.67	94.44	95.38	95.78
ITLDA	93.83	95.17	96.05	96.45

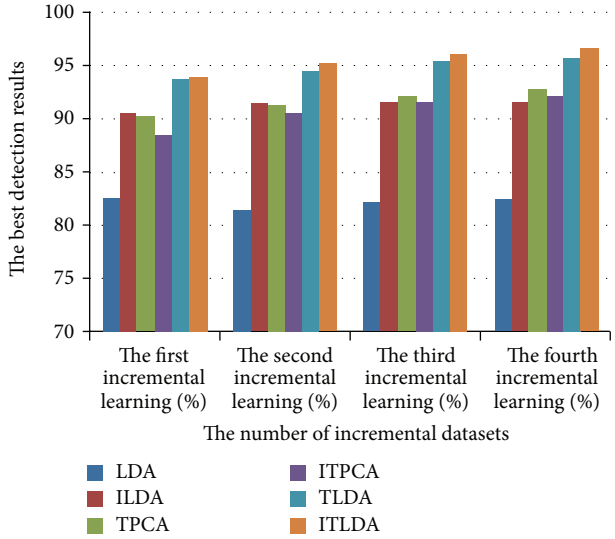


Figure 6: The comparison of the best detection results for different algorithms with incremental learning.

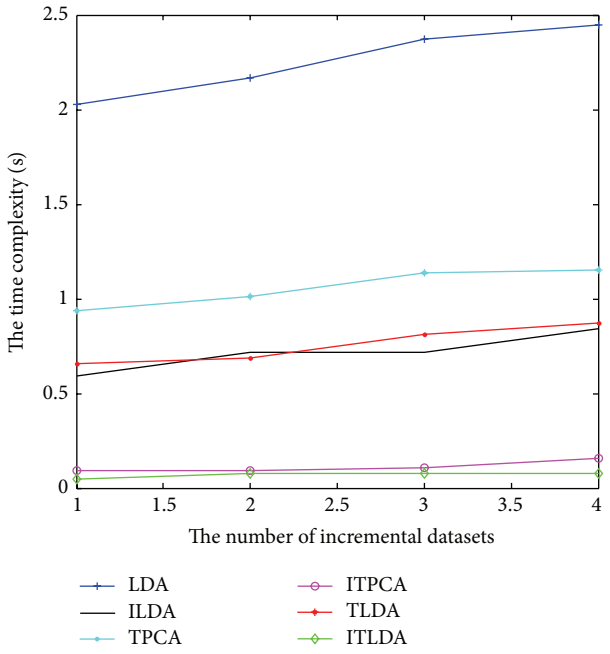


Figure 7: The comparison of time complexity.

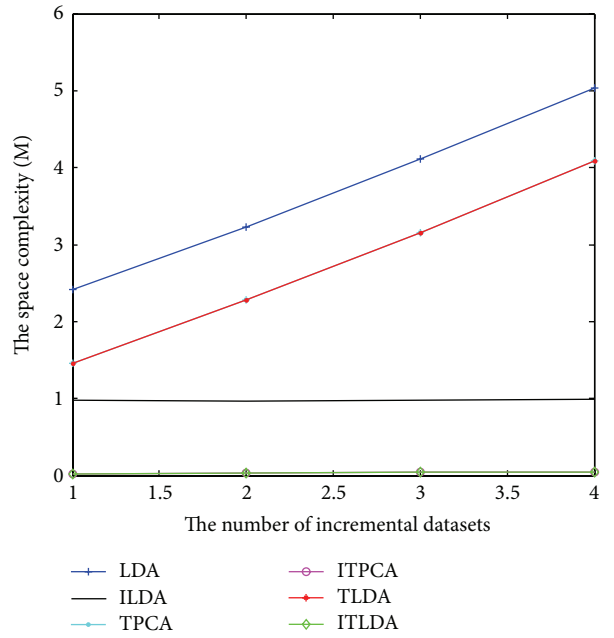


Figure 8: The comparison of space complexity.

apparently compared with nonincremental learning. Furthermore, since ITPCA and ITDA adopt tensor representation, they have lower time and space requirements than LDA.

5. Conclusions

In this paper, incremental tensor discriminant analysis (ITDA) is investigated. It adopts tensor representation to keep the structure information for high-dimensional images and introduces incremental learning to complete online learning. This paper also proves the relationship between ITDA and the graph framework theoretically. The facial detection experiments have shown that ITDA has better performance than TDA and is able to reduce the time and space complexity apparently.

Conflict of Interests

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

Acknowledgments

This present work has been funded with support from the Young Scientist Project of Chengdu University (no. 2013XJZ21), Project of Science and Technology Support Program of Sichuan Province, China, under Grant no. 2014GZ0013, and Project of Education Department of Sichuan Province, China, under Grant no. 13ZA0297.

References

[1] M. Murtaza, M. Sharif, M. Raza, and J. Shah, "Face recognition using adaptive margin fisher's criterion and linear discriminant analysis," *International Arab Journal of Information Technology*, vol. 11, no. 2, pp. 1–11, 2014.

[2] C. R. Rao, "The utilization of multiple measurements in problems of biological classification," *Journal of the Royal Statistical Society Series B (Methodological)*, vol. 10, no. 2, pp. 159–203, 1948.

[3] C. Liu, K. He, J.-L. Zhou, and C.-B. Gao, "Discriminant orthogonal rank-one tensor projections for face recognition," in *Intelligent Information and Database Systems*, vol. 6592 of *Lecture Notes in Computer Science*, pp. 203–211, Springer, Berlin, Germany, 2011.

[4] G.-F. Lu, Z. Lin, and Z. Jin, "Face recognition using discriminant locality preserving projections based on maximum margin criterion," *Pattern Recognition*, vol. 43, no. 10, pp. 3572–3579, 2010.

[5] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "Uncorrelated multilinear discriminant analysis with regularization and aggregation for tensor object recognition," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 103–123, 2009.

[6] J.-L. Minoi, C. E. Thomaz, and D. F. Gillies, "Tensor-based multivariate statistical discriminant methods for face applications," in *Proceedings of the International Conference on Statistics in Science, Business, and Engineering (ICSSBE '12)*, pp. 1–6, IEEE, Langkawi, Malaysia, September 2012.

[7] F. Nie, S. Xiang, Y. Song, and C. Zhang, "Extracting the optimal dimensionality for local tensor discriminant analysis," *Pattern Recognition*, vol. 42, no. 1, pp. 105–114, 2009.

[8] N. Tang, X. Z. Gao, and X. Li, "Tensor subclass discriminant analysis for radar target classification," *Electronics Letters*, vol. 48, no. 8, pp. 455–456, 2012.

[9] D. Tao, X. Li, X. Wu, and S. J. Maybank, "General tensor discriminant analysis and Gabor features for gait recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 10, pp. 1700–1715, 2007.

[10] S.-J. Wang, J. Yang, M.-F. Sun, X.-J. Peng, M.-M. Sun, and C.-G. Zhou, "Sparse tensor discriminant color space for face verification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 6, pp. 876–888, 2012.

[11] Z.-Z. Yu, C.-C. Jia, W. Pang, C.-Y. Zhang, and L.-H. Zhong, "Tensor discriminant analysis with multiscale features for action modeling and categorization," *IEEE Signal Processing Letters*, vol. 19, no. 2, pp. 95–98, 2012.

[12] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin, "Graph embedding and extensions: a general framework for dimensionality reduction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 40–51, 2007.

[13] A. Joseph, Y.-M. Jang, S. Ozawa, and M. Lee, "Extension of incremental linear discriminant analysis to online feature extraction under nonstationary environments," in *Neural Information Processing*, vol. 7664 of *Lecture Notes in Computer Science*, pp. 640–647, Springer, Berlin, Germany, 2012.

[14] G.-F. Lu, J. Zou, and Y. Wang, "Incremental complete LDA for face recognition," *Pattern Recognition*, vol. 45, no. 7, pp. 2510–2521, 2012.

[15] G.-F. Lu, J. Zou, and Y. Wang, "Incremental learning of complete linear discriminant analysis for face recognition," *Knowledge-Based Systems*, vol. 31, pp. 19–27, 2012.

[16] G.-F. Lu, J. Zou, and Y. Wang, "Incremental learning of discriminant common vectors for feature extraction," *Applied Mathematics and Computation*, vol. 218, no. 22, pp. 11269–11278, 2012.

[17] Q. Wang and L. Zhang, "Least squares online linear discriminant analysis," *Expert Systems with Applications*, vol. 39, no. 1, pp. 1510–1517, 2012.

[18] J. Sun, D. Tao, S. Papadimitriou, P. S. Yu, and C. Faloutsos, "Incremental tensor analysis: theory and applications," *ACM Transactions on Knowledge Discovery from Data*, vol. 2, no. 3, article 11, 2008.

[19] J.-G. Wang, E. Sung, and W.-Y. Yau, "Incremental two-dimensional linear discriminant analysis with applications to face recognition," *Journal of Network and Computer Applications*, vol. 33, no. 3, pp. 314–322, 2010.

[20] J. Wen, X. Gao, Y. Yuan, D. Tao, and J. Li, "Incremental tensor biased discriminant analysis: a new color-based visual tracking method," *Neurocomputing*, vol. 73, no. 4–6, pp. 827–839, 2010.

[21] P. N. Belhumeur, J. P. Hespanha, and D. Kriegman, "Eigenfaces vs. fisherfaces: recognition using class specific linear projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720, 1997.

[22] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "MPCA: multilinear principal component analysis of tensor objects," *IEEE Transactions on Neural Networks*, vol. 19, no. 1, pp. 18–39, 2008.

[23] C. Liu, T. Yan, W. Zhao et al., "Incremental tensor principal component analysis for handwritten digit recognition," *Mathematical Problems in Engineering*, vol. 2014, Article ID 819758, 10 pages, 2014.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 760403, 7 pages
http://dx.doi.org/10.1155/2015/760403

Research Article

Application of Z-Number Based Modeling in Psychological Research

Rafik Aliev[1] and Konul Memmedova[2]

[1]*Department of Computer Aided Design, Azerbaijan State Oil Academy, Azerbaijan*
[2]*Department of Psychological Counselling and Guidance, Near East University, 99138 Lefkoşa, Northern Cyprus, Mersin 10, Turkey*

Correspondence should be addressed to Konul Memmedova; konul.memmedova@neu.edu.tr

Received 5 December 2014; Revised 6 February 2015; Accepted 17 February 2015

Academic Editor: Rahib H. Abiyev

Pilates exercises have been shown beneficial impact on physical, physiological, and mental characteristics of human beings. In this paper, Z-number based fuzzy approach is applied for modeling the effect of Pilates exercises on motivation, attention, anxiety, and educational achievement. The measuring of psychological parameters is performed using internationally recognized instruments: Academic Motivation Scale (AMS), Test of Attention ($D2$ Test), and Spielberger's Anxiety Test completed by students. The GPA of students was used as the measure of educational achievement. Application of Z-information modeling allows us to increase precision and reliability of data processing results in the presence of uncertainty of input data created from completed questionnaires. The basic steps of Z-number based modeling with numerical solutions are presented.

1. Set of Problem

The investigation of the effect of physical activities on motivation, attention, anxiety, and educational performances of students is one of the important research areas in psychology. Hannaford in [1] noticed that "Thinking and learning are not all in our head. Our movements that not only express knowledge and facilitate greater cognitive function, they actually grow the brain as they increase in complexity. Our entire brain structure is intimately connected to and grown by the movement mechanism within our body." Recently some research works, demonstrating the achievement of the academic performances through physical activities, have been published in different sources. In [2] is given a review showing the effectiveness of aerobic, resistance, and multimodal exercise interventions on a wide range of outcome measures, including cognition, general physical function, mobility, strength, balance, flexibility, quality of life. It was shown [3] that the students with the highest fitness level performed better on standardized tests and students with the lowest fitness level performed lower in class grades. In [4–6] it was found that the physical activity improves a cognitive function, learning, and academic achievement. Salmon in [7] presented cross-sectional and longitudinal studies analyzing the effect of physical exercises on anxiety, depression, and sensitivity to stress. Increased physical activity therefore reduces premature mortality [8] and the establishment and maintenance of exercises' habits has become target for clinical psychologists [9].

The above research studies consider impact of psychological activities on performances of students using different psychological parameters. Considering and modeling the effect of physical activities on basic psychological parameters of humans acquire great importance. In this paper, we suggest the model that takes the parameters—motivation, attention, and anxiety—into account and improves educational achievement of students through Pilates exercises.

The structure that we have designed for modeling the effect of Pilates exercises on motivation, attention, anxiety, and educational achievement is presented in Figure 1.

There is broad agreement in the literature that the physical exercises and particularly Pilates exercises are associated

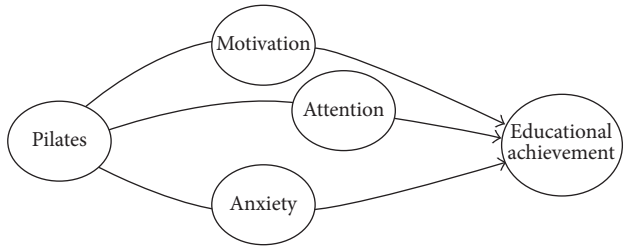


Figure 1

with the motivation, attention, anxiety, and educational achievement [10–15]. The classical statistical approach is used today for finding the relationship between Pilates and above mentioned psychological parameters. Application of statistical techniques allows estimating the relationship between input (effect of Pilates) and output (achievement) from probabilistic point of view. This approach embraces only statistical uncertainty, but not fuzzy uncertainty inherent in psychological phenomena. Motivation of the application of Z-number theory to solve the presented problem is as follows.

(1) The concepts in the human brain for perceiving, recognizing, and categorizing natural phenomena are vague and imprecise. The boundaries of these parameters (such as low, moderate, or high levels of anxiety, motivation, and attention) are not exactly defined. Therefore, claiming that this emerges from them also becomes vague.

(2) Z-number theory allows us to estimate the relationship between input and output by using the concept of fuzzy information and partial reliability.

(3) Z-number based approach enables us to use uncertainty measures to quantify the ambiguity associated with prediction of psychological variables.

We will start with short introduction of basic terms: Pilates, motivation, attention, and anxiety.

2. Introduction

2.1. Benefits of Pilates. In developing his method, Pilates (founder of Pilates exercises) combines both Eastern and the Western concepts [10] by including mental focus and specific breathing of yoga with the Western physical exercise systems.

The mind-body approach is a basic of Pilates principles: centering, concentration, control, precision, flow, and breath [10, 11].

Pilates has the following physical, psychological, and social impacts [11–15].

(1) Pilates exercise improves muscular strength, balance, posture, flexibility, and bone density and decreases back pain. The development of musculoskeletal fitness with long term resistance training is associated with enhanced cardiovascular function and musculoskeletal metabolism.

(2) Pilates increases a cognitive function—Pilates is different than many other forms of exercise because it requires the mind to pay attention to what you are doing. Research shows that when required to think about how you are moving, your brain cells grow at a faster rate and your nervous system creates better connections throughout your body.

(3) Pilates increases brain neurotransmitters, brain-derived neurotrophins that support neuronal differentiation and survival in the developing brain. Neurotrophins assure the survival of neurons in areas responsible for learning, memory, and higher thinking and, as an overall result, improve the overall quality of life.

(4) Pilates increases blood flow and oxygen flow to the brain and raises levels of serotonin and endorphins—all of which help to reduce stress, anxiety, and fatigue and improve mood, motivation, and achievement.

(5) Beyond academic achievement, many researchers connect Pilates to absenteeism, drop-out rate, and social communications of students.

2.2. Motivation. Motivation is defined as the process that initiates, guides, and maintains goal-oriented behaviors. It involves the biological, emotional, social, and cognitive forces that activate behavior. Motivation has been shown to positively influence study strategy and academic performance. Self-determination theory describes human motivation on a continuum categorized by amotivation, instincts, and extrinsic motivations [16].

Amotivation is the state of lacking the intention to act, whether resulting from not valuing the activity, not feeling capable of performing the task, or not expecting it to produce a desired outcome.

Extrinsic motivations are those that arise from outside of the individual and often involve rewards such as trophies, money, social recognition, or praise.

Intrinsic motivations are those that arise from within the individual, for the inherent satisfaction of the activity itself.

Pilates has effect on instinct motivation through the following mechanisms: improving total mood, body energy, self-esteem, psychological well-being, and vitality, reducing stress and anxiety, releasing certain neurotransmitters that alleviate physical and mental pain, and satisfying the basic psychological needs as competence, autonomy, and relatedness.

2.3. Anxiety. Anxiety is one of the major psychological variables which is considered as an important part of personality development. Anxiety is a psychological and physiological state characterized by somatic, emotional, cognitive, and behavioural components [17]. Anxiety generally helps in improving the performance of an individual. However, when anxiety becomes overwhelming, it may fall under the classification of anxiety disorder. This means anxiety should not cross its threshold value; otherwise it will reach up to an abnormal level.

American and European studies have found a negative correlation between anxiety and academic achievement.

Pilates has effect on anxiety through the following mechanisms: increasing body energy, sleep quality, attention, and concentration, releasing negatively thinking, improving blood and oxygen circulated to the brain, and relaxing muscles.

2.4. Attention. Attention was originally defined as processing one out of what seem several simultaneously possible objects or trains of thought. It implies withdrawal from some things in order to deal effectively with others [18].

Attention can be considered as a filter, in which many pieces of information come into the brain, but only one of these pieces of information is processed.

Pilates has effect on attention through the following mechanisms: concentration and precision are two main principles of Pilates; controlling a body movement by the brain; increasing sleep quality; relaxing of body; releasing negative thinking; decreasing the stress and anxiety.

3. Application of Fuzzy Logic and Z-Number Theory in Psychology Researches

The applications of fuzzy logic in psychology researches have been started since mid-1980 [19]. Averkin and Tarasov in [20] examined application of fuzzy modeling relation in psychology. Hesketh et al. considered [21] application of fuzzy graphical rating scale to the psychology. In [22] fuzzy logic based model of emotion is given. The role of fuzzy logic in psychological researches was examined in [23]. The researchers studied the relationship between motivation and anxiety using fuzzy logic and concluded that the fuzzy logic method has more advantages over statistical analysis to control uncertainties in data.

In this paper, we put a new approach for modeling of psychology processes through the Z-valuation concept. Z-information approach includes value of variable of interest and its reliability and has a major advantage for modeling such type of concepts. This approach enables us to use uncertainty measures to quantify the ambiguity associated with prediction of psychological parameters.

3.1. Preliminaries. A discrete Z-number [24] is an ordered pair $Z = (\widetilde{A}, \widetilde{B})$, where $\widetilde{A}$ is a discrete fuzzy number playing a role of a fuzzy constraint on values that a random variable X may take:

$$X \text{ is } \widetilde{A} \tag{1}$$

and $\widetilde{B}$ is a discrete fuzzy number with a membership function $\mu_{\widetilde{B}} : \{b_1, \ldots, b_n\} \rightarrow [0, 1]$, $\{b_1, \ldots, b_n\} \subset [0, 1]$, playing a role of a fuzzy constraint on the probability measure of $\widetilde{A}$:

$$P\left(\widetilde{A}\right) \text{ is } \widetilde{B}. \tag{2}$$

A concept of a discrete Z^+-number is closely related to the concept of a discrete Z-number. Given a discrete Z-number $Z = (\widetilde{A}, \widetilde{B})$, Z^+-number Z^+ is a pair consisting of a fuzzy number, $\widetilde{A}$, and a random number R:

$$Z^+ = \left(\widetilde{A}, R\right), \tag{3}$$

where $\widetilde{A}$ plays the same role as it does in a discrete Z-number $Z = (\widetilde{A}, \widetilde{B})$ and R plays the role of the probability distribution p, such that $P(\widetilde{A}) = \sum_{i=1}^{n} \mu_{\widetilde{A}}(x_i) p(x_i)$, $P(\widetilde{A}) \in \text{supp}(\widetilde{B})$.

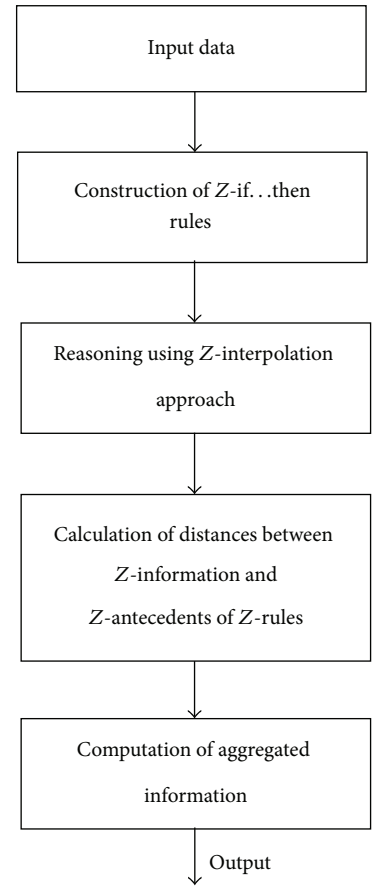


Figure 2: Flowchart diagram of Z-number modeling.

4. Problem Solving

Flowchart diagram of Z-number based modeling is given in Figure 2.

Input raw data is created from the questionnaires (tests of motivation, attention, and anxiety) completed by students.

This data is imprecise and involves uncertainty related with process of completing (filling) of questionnaires. The results of measuring are processed as fuzzy variables with the different fuzzy subsets.

The relationships between educational achievement and the above mentioned psychological variables are presented as *Z-if. . .then* rules (Table 1) by the following linguistic variables: H—high; L—low; M—medium; G—good; E—excellence; U—usually; P—plausible; R—rare.

Z-rule base concept plays pivotal role in economics, decision making, forecasting, and other human centric systems

Table 1: *Z-if...then* rules.

Number of rules	If			Then
	Motivation	Attention	Anxiety	Achievement
1	(L, U)	(L, U)	(H, U)	(L, U)
2	(L, U)	(M, U)	(M, U)	(M, U)
3	(L, U)	(M, U)	(L, U)	(G, U)
4	(L, U)	(H, U)	(L, U)	(G, U)
5	(L, U)	(H, U)	(L, U)	(E, U)
6	(L, U)	(L, U)	(M, U)	(L, U)
7	(L, U)	(H, U)	(M, U)	(G, U)
8	(M, U)	(H, U)	(L, U)	(E, P)
9	(M, U)	(M, U)	(M, U)	(G, U)
10	(M, U)	(M, U)	(M, U)	(M, U)
11	(M, U)	(M, U)	(H, U)	(L, U)
12	(M, U)	(H, U)	(L, U)	(G, U)
13	(M, U)	(H, U)	(L, U)	(E, U)
14	(M, U)	(H, U)	(H, U)	(M, U)
15	(M, U)	(H, U)	(M, U)	(M, U)
16	(M, U)	(H, U)	(M, U)	(G, U)
17	(M, U)	(H, U)	(M, U)	(E, U)
18	(H, U)	(H, U)	(L, U)	(E, R)
19	(H, U)	(H, U)	(L, U)	(G, U)
20	(H, U)	(H, U)	(M, U)	(E, U)
21	(H, U)	(H, U)	(M, U)	(G, U)
22	(H, U)	(M, U)	(M, U)	(G, U)
23	(H, U)	(M, U)	(L, U)	(G, U)
24	(H, U)	(M, U)	(L, U)	(M, U)
25	(H, U)	(L, U)	(H, U)	(L, U)
26	(H, U)	(L, U)	(M, U)	(M, U)
27	(H, U)	(L, U)	(L, U)	(L, U)
28	(H, U)	(L, U)	(L, U)	(M, U)

functioning in Z-information environment. The Z-rule base is complete when for all the possible observations there exists at least one rule whose Z-antecedent part overlaps the current antecedent Z-valuation, at least partially. Otherwise, the Z-rule base is incomplete. In case that there is incomplete (sparse) Z-rule base, the classical reasoning methods based on compositional rule of inference (Zadeh [25], Mamdani [26], and R. A. Aliev and R. R. Aliev [27]) or Takagi and Sugeno [28] reasoning approach are not so effective to adapt generating an output for the observation covered by none of the rules. Consequently, we will use inference techniques which in the lack of matching rules can perform an approximate reasoning, namely, Z-interpolated methods.

A problem of Z-interpolation is an interpolation of of fuzzy is given below [24, 29].

Given the following Z-rules:

$$\begin{aligned} &\text{If } X \text{ is } \left(A_{X,1}, B_{X,1}\right) \text{ then } Y \text{ is } \left(A_{Y,1}, B_{Y,1}\right), \\ &\text{If } X \text{ is } \left(A_{X,2}, B_{X,2}\right) \text{ then } Y \text{ is } \left(A_{Y,2}, B_{Y,2}\right), \\ &\qquad\vdots \\ &\text{If } X \text{ is } \left(A_{X,n}, B_{X,n}\right) \text{ then } Y \text{ is } \left(A_{Y,n}, B_{Y,n}\right) \end{aligned} \tag{4}$$

and the fact that

$$X \text{ is } \left(A_X, B_X\right), \tag{5}$$

find the Z-value of Y.

The idea underlying the suggested interpolation approach is that the ratio of distances between the conclusion and the consequent parts is identical to ones between the observation and the antecedent parts. For Z-rules interpolation we have

$$Z_Y = \frac{\sum_{i=1}^{n} \left(1/\operatorname{dist}\left(Z_X, Z_{X,i}\right)\right) Z_{Y,i}}{\sum_{k=1}^{n} \left(1/\operatorname{dist}\left(Z_X, Z_{X,i}\right)\right)}, \tag{6}$$

where dist is the distance between Z-numbers. As dist, method suggested in [29] can be used.

Let us consider the special case of the considered problem of Z-rules interpolation.

Given the Z-rules

$$\begin{aligned} &\text{If } X \text{ is } A_{X,1} \text{ then } Y \text{ is } \left(A_{Y,1}, B\right), \\ &\text{If } X \text{ is } A_{X,2} \text{ then } Y \text{ is } \left(A_{Y,2}, B\right), \\ &\qquad\vdots \\ &\text{If } X \text{ is } A_{X,n} \text{ then } Y \text{ is } \left(A_{Y,n}, B\right) \end{aligned} \tag{7}$$

and the fact that X is (A_X, B_X), find the value of Y.

For this case, formula (6) is reduced to

$$\begin{aligned} Z_Y &= \frac{\sum_{i=1}^{n} \left(1/\operatorname{dist}\left(Z_X, Z_{X,i}\right)\right) Z_{Y,i}}{\sum_{k=1}^{n} \left(1/\operatorname{dist}\left(Z_X, Z_{X,i}\right)\right)} \\ &= \frac{\sum_{i=1}^{n} \left(1/\operatorname{dist}\left(A_X, A_{X,i}\right)\right) \left(A_{Y,i}, B\right)}{\sum_{k=1}^{n} \left(1/\operatorname{dist}\left(A_X, A_{X,i}\right)\right)}. \end{aligned} \tag{8}$$

For example, using supremum metric d of fuzzy numbers will be

$$\begin{aligned} Z_Y &= \frac{\sum_{i=1}^{n} \left(1/\operatorname{dist}\left(A_X, A_{X,i}\right)\right) \left(A_{Y,i}, B\right)}{\sum_{k=1}^{n} \left(1/\operatorname{dist}\left(A_X, A_{X,i}\right)\right)} \\ &= \frac{\sum_{i=1}^{n} \left(1/d\left(A_X, A_{X,i}\right)\right) \left(A_{Y,i}, B\right)}{\sum_{k=1}^{n} \left(1/d\left(A_X, A_{X,i}\right)\right)}. \end{aligned} \tag{9}$$

Taking into account that $1/d(A_X, A_{X,i})$ is a scalar and applying the approach to multiplication of a Z-number by a scalar described in [30] we will have

$$Z_Y = \left(A_Y, B\right), \quad \text{with } A_Y = \frac{\sum_{i=1}^{n} \left(1/d\left(A_X, A_{X,i}\right)\right) A_{Y,i}}{\sum_{k=1}^{n} \left(1/d\left(A_X, A_{X,i}\right)\right)}. \tag{10}$$

Let the knowledge base of 26 Z-rules of the following form be given:

$$\begin{aligned} &\text{If } X_1 \text{ is } \widehat{Z}_{i1} = (A_{i1}, B_{i1}), \\ &\quad X_2 \text{ is } \widehat{Z}_{i2} = (A_{i2}, B_{i2}), \\ &\quad X_3 \text{ is } \widehat{Z}_{i3} = (A_{i3}, B_{i3}) \\ &\text{then } Y \text{ is } \widehat{Z}_{Y,i} = \left(A_{Y_i}, B_{Y_i}\right), \\ &\qquad i = 1, \ldots, 26. \end{aligned} \tag{11}$$

The considered Z-rules were described in terms of linguistic labels of A_{ij}, B_{ij}, given in Table 1.

Consider a problem of reasoning within the given Z-rules base by using Z-interpolation approach. Let the current input information be described by the following Z-numbers $\widehat{Z}_1 = (Z_{A_1}, Z_{B_1})$, $\widehat{Z}_2 = (Z_{A_2}, Z_{B_2})$, $\widehat{Z}_3 = (Z_{A_3}, Z_{B_3})$:

$$\begin{aligned} Z_{A_1} &= \frac{0.1}{0.3} + \frac{0.5}{0.35} + \frac{1}{0.4} + \frac{0.5}{0.45} + \frac{0.1}{0.5}, \\ Z_{B_1} &= \frac{0.1}{0.6} + \frac{0.5}{0.65} + \frac{1}{0.7} + \frac{0.5}{0.75} + \frac{0.1}{0.8}, \\ Z_{A_2} &= \frac{0.1}{0.2} + \frac{0.5}{0.25} + \frac{1}{0.3} + \frac{0.5}{0.35} + \frac{0.1}{0.4}, \\ Z_{B_2} &= \frac{0.1}{0.6} + \frac{0.5}{0.65} + \frac{1}{0.7} + \frac{0.5}{0.75} + \frac{0.1}{0.8}, \\ Z_{A_3} &= \frac{0.1}{0.62} + \frac{0.5}{0.67} + \frac{1}{0.7} + \frac{0.5}{0.73} + \frac{0.1}{0.8}, \\ Z_{B_3} &= \frac{0.1}{0.6} + \frac{0.5}{0.65} + \frac{1}{0.7} + \frac{0.5}{0.75} + \frac{0.1}{0.8}. \end{aligned} \tag{12}$$

Z-interpolation approach based reasoning consists of two main stages.

(1) For each rule compute distance D_i between the current input Z-information $\widehat{Z}_1 = (Z_{A_1}, Z_{B_1})$, $\widehat{Z}_2 = (Z_{A_2}, Z_{B_2})$, $\widehat{Z}_3 = (Z_{A_3}, Z_{B_3})$ and Z-antecedents of Z-rules base $\widehat{Z}_{i1} = (A_{i1}, B_{i1})$, $\widehat{Z}_{i2} = (A_{i2}, B_{i2})$, $\widehat{Z}_{i3} = (A_{i3}, B_{i3})$ as follows:

$$D_i = \sum_{j=1}^{3} D\left(\widehat{Z}_j, \widehat{Z}_{ij}\right), \tag{13}$$

where $D(\widehat{Z}_j, \widehat{Z}_{ij})$ is the supremum metric:

$$D\left(\widehat{Z}_j, \widehat{Z}_{ij}\right) = d_H\left(\widetilde{A}_j, \widetilde{A}_{ij}\right) + d_H\left(\widetilde{B}_j, \widetilde{B}_{ij}\right), \tag{14}$$

with

$$\begin{aligned} d_H\left(\widetilde{A}_j, \widetilde{A}_{ij}\right) &= \sup\left\{d_H\left(A_j{}^{\alpha}, A_{ij}{}^{\alpha}\right) \mid 0 < \alpha \le 1\right\}, \\ d_H\left(\widetilde{B}_j, \widetilde{B}_{ij}\right) &= \sup\left\{d_H\left(B_j{}^{\alpha}, B_{ij}{}^{\alpha}\right) \mid 0 < \alpha \le 1\right\}. \end{aligned} \tag{15}$$

Consider computation of D_i for 1st and 15th rules. Z-antecedents of the 1st rule are Z-numbers $\widehat{Z}_{11} = (Z_{A_{11}}, Z_{B_{11}})$, $\widehat{Z}_{12} = (Z_{A_{12}}, Z_{B_{12}})$, $\widehat{Z}_{13} = (Z_{A_{13}}, Z_{B_{13}})$:

$$\begin{aligned} Z_{A_{11}} &= \frac{1}{0.1} + \frac{0.75}{0.2} + \frac{0.5}{0.3} + \frac{0.25}{0.4} + \frac{0.1}{0.5}, \\ Z_{B_{11}} &= \frac{0.1}{0.7} + \frac{0.5}{0.75} + \frac{1}{0.8} + \frac{0.5}{0.85} + \frac{0.1}{0.9}, \\ Z_{A_{12}} &= \frac{1}{0.1} + \frac{0.75}{0.2} + \frac{0.5}{0.3} + \frac{0.25}{0.4} + \frac{0.1}{0.5}, \\ Z_{B_{12}} &= \frac{0.1}{0.7} + \frac{0.5}{0.75} + \frac{1}{0.8} + \frac{0.5}{0.85} + \frac{0.1}{0.9}, \\ Z_{A_{13}} &= \frac{0.1}{0.5} + \frac{0.25}{0.57} + \frac{0.5}{0.65} + \frac{0.75}{0.72} + \frac{1}{0.8}, \\ Z_{B_{13}} &= \frac{0.1}{0.7} + \frac{0.5}{0.75} + \frac{1}{0.8} + \frac{0.5}{0.85} + \frac{0.1}{0.9}. \end{aligned} \tag{16}$$

Thus, we need to compute $D_1 = \sum_{j=1}^{3} D(\widehat{Z}_j, \widehat{Z}_{1j})$ according to (13), where $D(\widehat{Z}_1, \widehat{Z}_{11})$, $D(\widehat{Z}_2, \widehat{Z}_{12})$, $D(\widehat{Z}_3, \widehat{Z}_{13})$ are computed on the base of (14). We have obtained the results:

$$\begin{aligned} D\left(\widehat{Z}_1, \widehat{Z}_{11}\right) &= d_H\left(\widetilde{A}_1, \widetilde{A}_{11}\right) + d_H\left(\widetilde{B}_1, \widetilde{B}_{11}\right) \\ &= 0.5 + 0.5 = 1, \\ D\left(\widehat{Z}_2, \widehat{Z}_{12}\right) &= 1, \\ D\left(\widehat{Z}_3, \widehat{Z}_{13}\right) &= 0.62. \end{aligned} \tag{17}$$

Thus, the distance for 1st rule is

$$D_1 = 2.62. \tag{18}$$

The inputs of the 15th rule $\widehat{Z}_{15,1} = (Z_{A_{15,1}}, Z_{B_{15,1}})$, $\widehat{Z}_{15,2} = (Z_{A_{15,2}}, Z_{B_{15,2}})$, $\widehat{Z}_{15,3} = (Z_{A_{15,3}}, Z_{B_{15,3}})$ are

$$\begin{aligned} Z_{A_{15,1}} &= \frac{1}{0.1} + \frac{0.75}{0.2} + \frac{0.5}{0.3} + \frac{0.25}{0.4} + \frac{0.1}{0.5}, \\ Z_{B_{15,1}} &= \frac{0.1}{0.7} + \frac{0.5}{0.75} + \frac{1}{0.8} + \frac{0.5}{0.85} + \frac{0.1}{0.9}, \\ Z_{A_{15,2}} &= \frac{1}{0.1} + \frac{0.75}{0.2} + \frac{0.5}{0.3} + \frac{0.25}{0.4} + \frac{0.1}{0.5}, \\ Z_{B_{15,2}} &= \frac{0.1}{0.7} + \frac{0.5}{0.75} + \frac{1}{0.8} + \frac{0.5}{0.85} + \frac{0.1}{0.9}, \\ Z_{A_{15,3}} &= \frac{0.1}{0.5} + \frac{0.25}{0.57} + \frac{0.5}{0.65} + \frac{0.75}{0.72} + \frac{1}{0.8}, \\ Z_{B_{15,3}} &= \frac{0.1}{0.7} + \frac{0.5}{0.75} + \frac{1}{0.8} + \frac{0.5}{0.85} + \frac{0.1}{0.9}. \end{aligned} \tag{19}$$

Analogously, we computed the distance for 15th rule as

$$\begin{aligned} D\left(\widehat{Z}_1, \widehat{Z}_{15,1}\right) &= 0.8, \\ D\left(\widehat{Z}_2, \widehat{Z}_{15,2}\right) &= 1, \end{aligned}$$

$$D\left(\widehat{Z}_3, \widehat{Z}_{15,3}\right) = 1.02,$$
$$D_{15} = 2.82. \tag{20}$$

The distances computed for the rest of the rules are

$$\begin{aligned}
&D_1 = 2.62, \quad D_7 = 3.02, \quad D_{13} = 2.9, \quad D_{19} = 3,\\
&D_{25} = 2.52, \quad D_2 = 2.92, \quad D_8 = 2.9, \quad D_{14} = 2.42,\\
&D_{20} = 2.92, \quad D_{26} = 2.52, \quad D_3 = 3, \quad D_9 = 2.72,\\
&D_{15} = 2.82, \quad D_{21} = 2.92, \quad D_4 = 3.1, \quad D_{10} = 2.72,\\
&D_{16} = 2.82, \quad D_{22} = 2.82, \quad D_5 = 3.1, \quad D_{11} = 2.32,\\
&D_{17} = 2.82, \quad D_{23} = 2.9, \quad D_6 = 3.02, \quad D_{12} = 2.9,\\
&D_{18} = 3, \quad D_{24} = 2.9.
\end{aligned} \tag{21}$$

(2) Computation of the aggregated output Z_Y for Z-rules base by using linear Z-interpolation are as follows:

$$Z_Y = \sum_{i=1}^{n} w_i Z_{Y,i}, \quad w_i = \frac{1}{D_i \sum_{k=1}^{n} (1/D_k)}. \tag{22}$$

Thus, we need to compute convex combination of outputs Z_{Y_i} of the rules base. The outputs of the Z-rules base are as follows:

$$\begin{aligned}
Z_{A_{Y_1}} &= \frac{1}{0.1} + \frac{0.75}{0.2} + \frac{0.5}{0.3} + \frac{0.25}{0.4} + \frac{0.1}{0.5},\\
Z_{B_{Y_1}} &= \frac{0.1}{0.6} + \frac{0.5}{0.65} + \frac{1}{0.7} + \frac{0.5}{0.75} + \frac{0.1}{0.8},\\
&\vdots\\
Z_{A_{Y_{15}}} &= \frac{0.1}{0.1} + \frac{0.5}{0.3} + \frac{1}{0.5} + \frac{0.5}{0.62} + \frac{0.1}{0.75},\\
Z_{B_{Y_{15}}} &= \frac{0.1}{0.1} + \frac{0.5}{0.15} + \frac{1}{0.2} + \frac{0.5}{0.25} + \frac{0.1}{0.3},\\
&\vdots
\end{aligned} \tag{23}$$

Therefore, the aggregated output Z_Y is defined as

$$Z_Y = 0.042 Z_{Y_1} + 0.037 Z_{Y_2} + \cdots + 0.038 Z_{Y_{26}} = \left(A_Y, B_Y\right). \tag{24}$$

We have obtained the following result:

$$\begin{aligned}
A_Y &= \frac{0.1}{0.395} + \frac{0.5}{0.518} + \frac{1}{0.627} + \frac{0.5}{0.644} + \frac{0.25}{0.656},\\
B_Y &= \frac{0.1}{0.7} + \frac{0.5}{0.75} + \frac{1}{0.8} + \frac{0.5}{0.85} + \frac{0.1}{0.9}.
\end{aligned} \tag{25}$$

In accordance with codebook we have the following. Achievement is "medium" with the reliability "usually."

The analysis shows that there are no significant differences between the mean of results obtained by conventional statistical method and Z-number based modeling. The standard deviations of the outputs estimated by statistical method and Z-number based modeling show significant differences of results. The less variance in the Z-number based modeling method is related with the ability of this method to control uncertainty in data.

5. Conclusion

In this paper we have suggested a new approach for modeling of the effect of the Pilates exercises on students' motivation, attention, anxiety, and educational achievement. The uncertainty of data related with cognitive measuring of psychological parameters and their partial reliability have been promoted for the first time application of "*Z-if…then* rules" for modeling the considered relationship.

We used an inference techniques for approximate reasoning based on Z-interpolation method suggested by Zadeh to embrace incomplete and lack of matching rules.

Computer simulation proves adequacy of the model.

Conflict of Interests

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

References

[1] C. Hannaford, *Smart Moves: Why Learning Is Not All in Your Head*, Creat River Books, Salt Lake City, Utah, USA, 2nd edition, 2005.

[2] B. B. Fox, B. Hodgkinson, and D. Parker, "The effects of physical exercise on functional performance, quality of life, cognitive impairment and physical activity levels for older adults aged 65 years and older with a diagnosis of dementia: a systematic review," *The JBI Database of Systematic Reviews and Implementation Reports*, 2014.

[3] D. P. Coe, J. M. Pivarnik, C. J. Womack, M. J. Reeves, and R. M. Malina, "Health-related fitness and academic achievement in middle school students," *Journal of Sports Medicine and Physical Fitness*, vol. 52, no. 6, pp. 654–660, 2012.

[4] J. U. Edwards, L. Mauch, and M. R. Winkelman, "Relationship of nutrition and physical activity behaviors and fitness measures to academic performance for sixth graders in a Midwest city school district," *Journal of School Health*, vol. 81, no. 2, pp. 65–73, 2011.

[5] T. J. H. Keeley and K. R. Fox, "The impact of physical activity and fitness on academic achievement and cognitive performance in children," *International Review of Sport and Exercise Psychology*, vol. 2, no. 2, pp. 198–214, 2009.

[6] P. D. Tomporowski, C. L. Davis, P. H. Miller, and J. A. Naglieri, "Exercise and children's intelligence, cognition, and academic achievement," *Educational Psychology Review*, vol. 20, no. 2, pp. 111–131, 2008.

[7] P. Salmon, "Effects of physical exercise on anxiety, depression, and sensitivity to stress: a unifying theory," *Clinical Psychology Review*, vol. 21, no. 1, pp. 33–61, 2001.

[8] P. H. Fentem, "Benefits of exercise in health and disease," *British Medical Journal*, vol. 308, no. 6939, pp. 1291–1295, 1994.

[9] R. S. Paffenbarger Jr. and R. T. Hyde, "Exercise adherence, coronary heart disease and longevity," in *Exercise Adherence: Its Impact on Public Health*, R. K. Dishman, Ed., pp. 41–73, Human Kinetics Books, Champaign, Ill, USA, 1988.

[10] J. Pilates and J. Robbins, *Pilates' Return to Life Through Contrology. Revised Edition for the 21st Century*, 2012, (Originally published by Joseph Pilates in 1945).

[11] A. Ungaro, *Pilates: Body in Motion*, Dorling Kindersley Publishing, London, UK, 2002.

[12] K. Rodenroth, *A study of the relationship between physical fitness and academic performance [Partial fulfillment of the requirements for the degree doctor of education]*, Liberty University, 2010.

[13] A. R. Aladro-Gonzalvo, M. Machado-Díaz, J. Moncada-Jiménez, J. Hernández-Elizondo, and G. Araya-Vargas, "The effect of Pilates exercises on body composition: a systematic review," *Journal of Bodywork and Movement Therapies*, vol. 16, no. 1, pp. 109–114, 2012.

[14] K. Caldwell, M. Harrison, M. Adams, and N. Travis Triplett, "Effect of Pilates and taiji quan training on self-efficacy, sleep quality, mood, and physical performance of college students," *Journal of Bodywork and Movement Therapies*, vol. 13, no. 2, pp. 155–163, 2009.

[15] C. H. Hillman, K. I. Erickson, and A. F. Kramer, "Be smart, exercise your heart: exercise effects on brain and cognition," *Nature Reviews Neuroscience*, vol. 9, no. 1, pp. 58–65, 2008.

[16] E. L. Deci and R. M. Ryan, "Self-determination theory: a macrotheory of human motivation, development, and health," *Canadian Psychology*, vol. 49, no. 3, pp. 182–185, 2008.

[17] M. S. Chapell, Z. B. Blanding, M. Takahashi et al., "Test anxiety and academic performance in undergraduate and graduate students," *Journal of Educational Psychology*, vol. 97, no. 2, pp. 268–274, 2005.

[18] P. Castle and S. Buckler, "What Was I Saying? Concentration and Attention," http://www.sagepub.com/upm-data/28824_02_Castle_&_Buckler_Ch_02.pdf.

[19] M. Smithson, "Applications of fuzzy set concepts to behavioral sciences," *Mathematical Social Sciences*, vol. 2, no. 3, pp. 257–274, 1982.

[20] A. N. Averkin and V. B. Tarasov, "The fuzzy modelling relation and its application in psychology and artificial intelligence," *Fuzzy Sets and Systems*, vol. 22, no. 1-2, pp. 3–24, 1987.

[21] T. Hesketh, R. Pryor, and B. Hesketh, "Application of a computerized fuzzy graphic rating scale to the psychological measurement of individual differences," *International Journal of Man-Machine Studies*, vol. 29, no. 1, pp. 21–35, 1988.

[22] M. S. El-Nasr, J. Yen, T. R. Loerger, and Flami, *Fuzzy Logic Adaptive Model of Emotions Autonomous Agents and Multi-Agent Systems*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000.

[23] G. S. Kushwaha and S. Kumar, "Role of the fuzzy system in psychological research," *Europe's Journal of Psychology*, vol. 5, no. 2, pp. 123–134, 2009.

[24] L. A. Zadeh, "A note on Z-numbers," *Information Sciences*, vol. 181, no. 14, pp. 2923–2932, 2011.

[25] L. A. Zadeh, "Fuzzy sets," *Information and Computation*, vol. 8, pp. 338–353, 1965.

[26] E. H. Mamdani, "Applications of fuzzy logic to approximate reasoning using linguistic systems," *IEEE Transactions on Computers*, vol. 26, no. 12, pp. 1182–1191, 1977.

[27] R. A. Aliev and R. R. Aliev, *Soft Computing and Its Application*, World Scientific, River Edge, NJ, USA, 2001.

[28] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 15, no. 1, pp. 116–132, 1985.

[29] L. T. Kóczy and K. Hirota, "Approximate reasoning by linear rule interpolation and general approximation," *International Journal of Approximate Reasoning*, vol. 9, no. 3, pp. 197–225, 1993.

[30] R. A. Aliev, O. H. Huseynov, R. R. Aliyev, and A. A. Alizadeh, *The Arithmetic of Z-Numbers, Theory and Applications*, Word Scientific, 2015.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 735060, 9 pages
http://dx.doi.org/10.1155/2015/735060

Research Article

Predictive Modeling in Race Walking

Krzysztof Wiktorowicz,[1] Krzysztof Przednowek,[2] Lesław Lassota,[2] and Tomasz Krzeszowski[1]

[1]*Faculty of Electrical and Computer Engineering, Rzeszów University of Technology, 35-959 Rzeszów, Poland*
[2]*Faculty of Physical Education, University of Rzeszów, 35-959 Rzeszów, Poland*

Correspondence should be addressed to Krzysztof Wiktorowicz; kwiktor@prz.edu.pl

Received 15 December 2014; Accepted 18 June 2015

Academic Editor: Okyay Kaynak

This paper presents the use of linear and nonlinear multivariable models as tools to support training process of race walkers. These models are calculated using data collected from race walkers' training events and they are used to predict the result over a 3 km race based on training loads. The material consists of 122 training plans for 21 athletes. In order to choose the best model leave-one-out cross-validation method is used. The main contribution of the paper is to propose the nonlinear modifications for linear models in order to achieve smaller prediction error. It is shown that the best model is a modified LASSO regression with quadratic terms in the nonlinear part. This model has the smallest prediction error and simplified structure by eliminating some of the predictors.

1. Introduction

The level of today's high-performance sport is very high and very even. Both coaches and competitors are forced to search for and use newer and sometimes innovative solutions in the process of sports training [1]. A solution supporting this process may be the application of various types of regression models.

Prediction in sport concerns many aspects including the prediction of performance results [2, 3] or predicting sporting talent [4, 5]. Models predicting results in sport, taking into account the seasonal statistics of each team, were also constructed [6]. The application of predictive models in athletics was described by Maszczyk et al. [2], where the regression was used to predict results in a javelin throw. These models were applied to support the choice and selection of prospective javelin throwers.

Prediction of sports results using linear regression was also presented in the work by Przednowek and Wiktorowicz [7]. A linear predictive model, implemented by ridge regression, was applied to predict the outcomes of a walking race after the immediate preparation phase. As input for the model, the basic somatic features (height and weight) and training loads (training components) for each day of training were provided, and the output was the result expected over a distance of 5 km. In addition to linear models, artificial neural networks, whose parameters were specified in cross-validation, were also used to implement this task.

In the paper by Drake and James [8], the regressions estimating the results over distances of 5, 10, 20, and 50 km and the levels of the selected physiological parameters (e.g., VO_2max) were presented. The regressions applied were the classical linear models, and the R^2 criterion was chosen for the quality evaluation. This study included 23 women and 45 men. The amount of collected data was different depending on the task and ranged from 21 to 68 records.

A nonlinear regression equation to predict the maximum aerobic capacity of footballers was proposed by Chatterjee et al. [9]. The data came from 35 young players aged from 14 to 16. The experiment was to verify the use of the 20 m MST (Multistage Shuttle Run Test) in evaluating the performance of VO_2max. The talent of young hockey players was identified by Roczniok et al. [5] using a regression equation. The research involved 60 boys aged between 15 and 16, who attended selection camps. The applied regression model classified candidates for future training, based on selected parameters of the players. The logistic regression was used in the model as the classification method.

The nonlinear predictive models used in sport are also based on the selected methods of "data mining" [10]. Among them, an important role is played by fuzzy logic expert systems. Papić et al. [4] described practical application of such a system. The proposed system was based on knowledge of experts in the field of sport, as well as the data obtained as a result of motor tests. The model suggested the most suitable sport and it was designed to search for prospective sports talents.

The application of fuzzy modeling techniques in sports prediction was also presented by Mężyk and Unold [11]. The goal of their paper was to find the rules that can express swimmer's feelings the day after in-water training. The data was collected for two months among competitors practicing swimming. The swimmers were characterized by a good level of sports attainment (2nd sport class). The material obtained consisted of 12 attributes, and the total number of models was 480, out of which 136 were used in the final stage. The authors proved that their method was characterized by better predictive ability than the traditional methods of classification.

Other papers also concern the use of artificial neural networks in sports prediction [6]. Neural models are used to analyze the effectiveness of the training of swimmers, to identify handball players' tactics, or to predict sporting talent [12]. Many studies present the application of neural networks in various aspects of sports training [13–15]. These models support the planning of training loads, practice control, or the selection of sports.

An approach developed by the authors is the construction of models performing the task of predicting the results achieved by a competitor in the proposed sports training. This allows for the proper selection of training components and thus supports the achievement of the desired result. The aim of this study is to determine the effectiveness of selected linear and nonlinear models in predicting the outcome in a 3-kilometer walking race for the proposed training. The research hypothesis of the paper is stated as follows: the prediction error of 3 kilometers' result in race walking for nonlinear models can be smaller than for linear models.

The paper is organized as follows. In Section 2, the training data of the race walkers recorded during annual training cycle is described. Section 3 contains the methods used to build the linear and nonlinear predictive models, including ordinary least squares regression, regularized methods, that is, ridge, LASSO, and elastic net regressions, nonlinear least squares regression, and artificial neural networks as multilayer perceptron and radial basis function network. In Section 3, the criterion used to evaluate the performance of the models, calculated using mean square error in the process of cross-validation, is also defined. Section 4 describes the procedures used for building models and their evaluation in *R* language and STATISTICA software. The obtained results are analyzed and discussed in Section 5. Finally, in Section 6, the performed work is concluded.

2. Material

The predictive models were built using the training data of athletes practising race walking. The analysis involved a group of colts and juniors from Poland. Among the competitors were the finalists in the Polish Junior Indoor Championships and the Polish Junior Championships. The data of race walkers was recorded during the 2011-2012 season in the form of training means and training loads. The training mean is the type of work performed while the training load is the amount of work at a particular intensity done by an athlete during exercise [1]. In the material, which has been collected, 11 means of training were distinguished. The material was drawn from the annual training cycle for the following four phases: transition, general preparation, special preparation, and starting phase. The training data has the form of sums of training loads completed in one month of the chosen training phase. The material included 122 training patterns made by 21 race walkers.

Control of the training process in race walking requires different tests of physical fitness at every training level. Because this research concerns the competitors in colt and junior categories, thus in order to determine a unified criterion of the level of training, a result for 3000 m race walking was used. The choice of the distance of 3000 m is valid because this is the indoor walking competition.

The description of the variables under consideration and their basic statistics are presented in Table 1. The variables are as follows: arithmetic mean of $\overline{x}$, minimum value $x_{\min}$, maximum value $x_{\max}$, standard deviation SD, and coefficient of variation $V = \mathrm{SD}/\overline{x} \cdot 100\%$. The qualitative variables are X_1, X_2, X_3, X_4, which take their values from the set $\{0, 1\}$. The other variables, that is, $X_5, \ldots, X_{18}$, are quantitative variables. If the value at inputs X_1, X_2, X_3 is 0, it means that the transitional period is considered. Setting the value 1 on one of the inputs X_1, X_2, X_3, it means the training period is selected. The variable X_4 represents the gender of the competitor, where the value 0 denotes a female, while the value 1 denotes a male, and the age is represented by X_5. Basic somatic features of race walkers such as weight and height are presented in the form of BMI (X_6) expressed by the formula

$$\mathrm{BMI} = \frac{M}{H^2}\ \left[\mathrm{kg/m^2}\right], \tag{1}$$

where M is the body weight [kg] and H is the body height [m]. The variable X_7 denotes the current result over 3 km in seconds. Training loads are characterized by the following variables: running exercises (X_8), walking with different levels of intensity (X_9, X_{10}, X_{11}), exercises forming different types of endurance (X_{12}, X_{13}, X_{14}), exercises forming techniques (X_{15}), exercises forming muscle strength (X_{16}), exercises forming general fitness (X_{17}), and warming up exercises (X_{18}).

An example of data used for building the model has the form

$$\begin{aligned} \mathbf{x}_5 = [&0, 1, 0, 0,\\ &23, 22.09, 800, 32, 400, 112, 20, 16, 32.4, 48, 8,\\ &280, 640, 400], \\ y_5 = 800.& \end{aligned} \tag{2}$$

Table 1: The variables and their basic statistics.

Variable	Description	$\overline{x}$	$x_{\min}$	$x_{\max}$	SD	V [%]
Y	Result over 3 km [s]	936.9	780	1155	78.4	8.4
X_1	General preparation phase	—	—	—	—	—
X_2	Special preparation phase	—	—	—	—	—
X_3	Starting phase	—	—	—	—	—
X_4	Competitor's gender	—	—	—	—	—
X_5	Competitor's age [years]	18.9	14	24	3.0	15.6
X_6	BMI (body mass index) [kg/m^2]	19.3	16.4	22.1	1.7	8.7
X_7	Current result over 3 km [s]	962.6	795	1210	87.7	9.1
X_8	Overall running endurance [km]	30.9	0	56	10.6	34.4
X_9	Overall walking endurance in the 1st intensity range [km]	224.6	57	440	96.1	42.8
X_{10}	Overall walking endurance in the 2nd intensity range [km]	53.2	0	120	34.6	65.1
X_{11}	Overall walking endurance in the 3rd intensity range [km]	7.9	0	30	9.4	119.7
X_{12}	Short tempo endurance [km]	8.9	0	24	5	56.0
X_{13}	Medium tempo endurance [km]	8.3	0	32.4	8.6	103.2
X_{14}	Long tempo endurance [km]	12.9	0	56	16.1	125.0
X_{15}	Exercises forming technique (rhythm) of walking [km]	4.4	0	12	4.2	96.0
X_{16}	Exercises forming muscle strength [min]	90.2	0	360	104.8	116.3
X_{17}	Exercises forming general fitness [min]	522.0	120	720	109.9	21.0
X_{18}	Universal exercises (warm up) [min]	317.3	150	420	72.5	22.8

The vector $\mathbf{x}_5$ represents a 23-year-old race walker with BMI = 22.09 kg/m^2, who completes training in the special preparation phase. The result both before and after the training was the same and is equal to 800 s.

3. Methods

In this study, two approaches were considered. The first approach was based on white box models realized by modern regularized methods. These models are interpretable because their structure and parameters are known. The second approach was based on black box models realized by artificial neural networks.

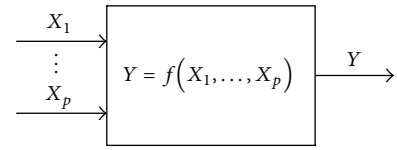


Figure 1: A diagram of a system with multiple inputs and one output.

3.1. Constructing Regression Models. Consider a multivariable regression model with the inputs (*predictors* or *regressors*) X_j, $j = 1, \ldots, p$, and one output (*response*) Y shown in Figure 1. We assume that the model is linear and has the form

$$\begin{aligned}\widehat{Y} &= w_0 + X_1 w_1 + \cdots + X_p w_p \\ &= w_0 + \sum_{j=1}^{p} X_j w_j,\end{aligned} \tag{3}$$

where $\widehat{Y}$ is the estimated response and w_0, w_j are unknown weights of the model. The weight w_0 is called *constant term* or *intercept*. Furthermore, we assume that the data is standardized and centered and the model can be simplified to the form (see, e.g., [16])

$$\begin{aligned}\widehat{Y} &= X_1 w_1 + \cdots + X_p w_p \\ &= \sum_{j=1}^{p} X_j w_j.\end{aligned} \tag{4}$$

Observations are written as pairs $(\mathbf{x}_i, y_i)$, where $\mathbf{x}_i = [x_{i1}, \ldots, x_{ip}]$, $i = 1, \ldots, n$, x_{ij} is the value of the jth predictor in the ith observation, and y_i is the value of the response in the ith observation. Based on formula (4), the ith observation can be expressed as

$$\begin{aligned}\widehat{y}_i &= x_{i1} w_1 + \cdots + x_{ip} w_p \\ &= \sum_{j=1}^{p} x_{ij} w_j = \mathbf{x}_i \mathbf{w},\end{aligned} \tag{5}$$

where $\mathbf{w} = [w_1, \ldots, w_p]^T$. Introducing matrix $\mathbf{X}$ in the form of

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} \tag{6}$$

formula (5) can be written as

$$\widehat{\mathbf{y}} = \mathbf{X}\mathbf{w}, \tag{7}$$

where $\widehat{\mathbf{y}} = [\widehat{y}_1, \ldots, \widehat{y}_n]^T$.

In order to construct regression models, an error (residual) is introduced as the difference between the real value y_i and the estimated value $\widehat{y}_i$ in the form of

$$e_i = y_i - \widehat{y}_i = y_i - \sum_{j=1}^{p} x_{ij} w_j = y_i - \mathbf{x}_i \mathbf{w}. \tag{8}$$

Using matrix (6), the error can be written as

$$\mathbf{e} = \mathbf{y} - \widehat{\mathbf{y}} = \mathbf{y} - \mathbf{X}\mathbf{w}, \tag{9}$$

where $\mathbf{e} = [e_1, \ldots, e_n]^T$ and $\mathbf{y} = [y_1, \ldots, y_n]^T$.

Denoting by $J(\mathbf{w}, \cdot)$ the *cost function*, the problem of finding the optimal estimator can be formulated as to minimize the function $J(\mathbf{w}, \cdot)$, which means solving the problem

$$\widehat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}} \left(J\left(\mathbf{w}, \cdot\right) \right), \tag{10}$$

where $\widehat{\mathbf{w}}$ is the vector of solutions.

Depending on the function $J(\mathbf{w}, \cdot)$, different regression models can be obtained. In this paper, the following models are considered: ordinary least squares regression (OLS), ridge regression, LASSO (least absolute shrinkage and selection operator), elastic net regression (ENET), and nonlinear least squares regression (NLS).

3.2. Linear Regressions. In OLS regression (see, e.g., [16–18]) the model is calculated by minimizing the sum of squared errors:

$$\begin{aligned} J\left(\mathbf{w}\right) &= \mathbf{e}^T \mathbf{e} \\ &= \left(\mathbf{y} - \mathbf{X}\mathbf{w}\right)^T \left(\mathbf{y} - \mathbf{X}\mathbf{w}\right) \\ &= \left\| \mathbf{y} - \mathbf{X}\mathbf{w} \right\|_2^2, \end{aligned} \tag{11}$$

where $\| \cdot \|_2$ denotes the Euclidean norm (L_2). Minimizing the cost function (11), which is the quadratic function of $\mathbf{w}$, we get the following solution:

$$\widehat{\mathbf{w}} = \left(\mathbf{X}^T \mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{y}. \tag{12}$$

It should be noted that solution (12) does not exist if the matrix $\mathbf{X}^T\mathbf{X}$ is singular (due to correlated predictors or if $p > n$). In this case, different methods of regularization, including the previously mentioned ridge, LASSO, and elastic net regressions, can be used.

In ridge regression by Hoerl and Kennard [19], the cost function includes a penalty and has the form

$$\begin{aligned} J\left(\mathbf{w}, \lambda\right) &= \mathbf{e}^T \mathbf{e} + \lambda \mathbf{w}^T \mathbf{w} \\ &= \left(\mathbf{y} - \mathbf{X}\mathbf{w}\right)^T \left(\mathbf{y} - \mathbf{X}\mathbf{w}\right) + \lambda \mathbf{w}^T \mathbf{w} \\ &= \left\| \mathbf{y} - \mathbf{X}\mathbf{w} \right\|_2^2 + \lambda \left\| \mathbf{w} \right\|_2^2. \end{aligned} \tag{13}$$

The parameter $\lambda \geq 0$ determines the size of the penalty: for $\lambda > 0$, the model is penalized, for $\lambda = 0$, ridge regression reduces to OLS regression. Solving problem (10) for ridge regression, we get

$$\widehat{\mathbf{w}} = \left(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I}\right)^{-1} \mathbf{X}^T \mathbf{y}, \tag{14}$$

where $\mathbf{I}$ is the identity matrix with the size of $p \times p$. Because the diagonal of the matrix $\mathbf{X}^T\mathbf{X}$ is increased by a positive constant, the matrix $\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}$ is invertible and the problem becomes nonsingular.

In LASSO regression by Tibshirani [20], similarly to ridge regression, the penalty is added to the cost function, where the L_1-norm (the sum of absolute values) is used:

$$\begin{aligned} J\left(\mathbf{w}, \lambda\right) &= \mathbf{e}^T \mathbf{e} + \lambda \mathbf{z}^T \mathbf{w} \\ &= \left(\mathbf{y} - \mathbf{X}\mathbf{w}\right)^T \left(\mathbf{y} - \mathbf{X}\mathbf{w}\right) + \lambda \mathbf{z}^T \mathbf{w} \\ &= \left\| \mathbf{y} - \mathbf{X}\mathbf{w} \right\|_2^2 + \lambda \left\| \mathbf{w} \right\|_1, \end{aligned} \tag{15}$$

where $\mathbf{z} = [z_1, \ldots, z_p]^T$, $z_j = \operatorname{sgn}(w_j)$, and $\| \cdot \|_1$ denotes the Manhattan norm (L_1). Because problem (10) is not linear in relation to $\mathbf{y}$ (due to the use of L_1-norm), the solution cannot be obtained in the compact form as in ridge regression. The most popular algorithm used in this case is the LARS algorithm (least angle regression) by Efron et al. [21].

In elastic net regression by Zou and Hastie [22], the features of ridge and LASSO regressions are combined. The cost function in the so-called *naive elastic net* has the form of

$$\begin{aligned} J\left(\mathbf{w}, \lambda_1, \lambda_2\right) &= \mathbf{e}^T \mathbf{e} + \lambda_1 \mathbf{z}^T \mathbf{w} + \lambda_2 \mathbf{w}^T \mathbf{w} \\ &= \left(\mathbf{y} - \mathbf{X}\mathbf{w}\right)^T \left(\mathbf{y} - \mathbf{X}\mathbf{w}\right) + \lambda_1 \mathbf{z}^T \mathbf{w} \\ &\quad + \lambda_2 \mathbf{w}^T \mathbf{w} \\ &= \left\| \mathbf{y} - \mathbf{X}\mathbf{w} \right\|_2^2 + \lambda_1 \left\| \mathbf{w} \right\|_1 + \lambda_2 \left\| \mathbf{w} \right\|_2^2. \end{aligned} \tag{16}$$

To solve the problem, Zou and Hastie [22] proposed the LARS-EN algorithm, which was based on the LARS algorithm developed for LASSO regression. They used the fact that elastic net regression reduces to LASSO regression for the augmented data set $(\mathbf{X}^*, \mathbf{y}^*)$.

3.3. Nonlinear Regressions. To take into account the nonlinearity in the models, we can apply the transformation of predictors or use nonlinear regression. In this paper, the latter solution is applied.

In OLS regression, the model is described by formula (5), while in more general nonlinear regression the relationship between the output and the predictors is expressed by a certain nonlinear function $f(\cdot)$ in the form of

$$\widehat{y}_i = f\left(\mathbf{x}_i, \mathbf{w}\right). \tag{17}$$

In this case, the cost function $J(\mathbf{w})$ is formulated as

$$\begin{aligned} J\left(\mathbf{w}\right) &= \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(y_i - \widehat{y}_i\right)^2 \\ &= \sum_{i=1}^{n} \left(y_i - f\left(\mathbf{x}_i, \mathbf{w}\right)\right)^2. \end{aligned} \tag{18}$$

Since the minimization of function (18) is associated with solving nonlinear equations, numerical optimization is used in this case. The main problem connected with the construction of nonlinear models is the choice of the appropriate function $f(\cdot)$.

3.4. Artificial Neural Networks. Artificial neural networks (ANNs) were also used for building predictive models. Two types of ANNs were implemented: a multilayer perceptron (MLP) and networks with radial basis function (RBF) [18].

The MLP network is the most common type of neural models. The calculation of the output in 3-layer multiple-input-one-output network is performed in *feed-forward* architecture. In the first step, m linear combinations, or the so-called *activations*, of the input variables are constructed as

$$a_k = \sum_{j=1}^{p} x_j w_{kj}^{(1)}, \tag{19}$$

where $k = 1, \ldots, m$ and $w_{kj}^{(1)}$ denotes the weights for the first layer. From the activations a_k, using a nonlinear *activation function* $h(\cdot)$, hidden variables z_k are calculated as

$$z_k = h\left(a_k\right). \tag{20}$$

The function $h(\cdot)$ is usually chosen as logistic or "tanh" function. The hidden variables are used next to calculate the *output activation*

$$a = \sum_{k=1}^{m} z_k w_k^{(2)}, \tag{21}$$

where $w_k^{(2)}$ are weights for the second layer. Finally, the output of the network is calculated using an activation function $\sigma(\cdot)$ in the form of

$$y = \sigma\left(a\right). \tag{22}$$

For regression problems, the function $\sigma(\cdot)$ is chosen as identity function, and so we obtain $y = a$. The MLP network utilizes iterative supervised learning known as *error back-propagation* for training the weights. This method is based on *gradient descent* applied to the sum of squares function. To avoid the problem with *overtraining* the network, the number m of hidden neurons, which is a free parameter, should be determined to give the best predictive performance.

In the RBF network, the concept of radial basis function is used. Linear regression (5) is extended by linear combinations of nonlinear functions of the inputs in the form of

$$\widehat{y}_i = \sum_{j=1}^{p} \phi_j\left(x_{ij}\right) w_j = \boldsymbol{\varphi}\left(\mathbf{x}_i\right)\mathbf{w}, \tag{23}$$

where $\boldsymbol{\varphi} = [\phi_1, \ldots, \phi_p]^T$ is a vector of *basis functions*. Using nonlinear basis functions, we get a nonlinear model, which is, however, a linear function of parameters w_j. In the RBF network, the hidden neurons perform a radial basis function whose value depends on the distance from selected center c_j:

$$\boldsymbol{\varphi}\left(\mathbf{x}_i, \mathbf{c}\right) = \boldsymbol{\varphi}\left(\left\|\mathbf{x}_i - \mathbf{c}\right\|\right), \tag{24}$$

where $\mathbf{c} = [c_1, \ldots, c_p]$ and $\|\cdot\|$ is usually the Euclidean norm. There are many possible choices for the basis functions, but the most popular is Gaussian function. It is known that RBF network can exactly interpolate any continuous function; that is, the function passes exactly through every data point. In this case, the number of hidden neurons is equal to the number of observations and the values of coefficients w_j are found by simple standard inversion technique. Such a network matches the data exactly, but it has poor predictive ability because the network is overtrained.

3.5. Choosing the Model. In this paper, the best predictive model is chosen using leave-one-out cross-validation (LOOCV) method [23], in which the number of tests is equal to the number of data n and one pair $(\mathbf{x}_i, y_i)$ creates a testing set. The quality of the model is evaluated by means of the square root of the mean square error ($\mathrm{RMSE}_{\mathrm{CV}}$) defined as

$$\begin{aligned} \mathrm{MSE}_{\mathrm{CV}} &= \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \widehat{y}_{-i}\right)^2, \\ \mathrm{RMSE}_{\mathrm{CV}} &= \sqrt{\mathrm{MSE}_{\mathrm{CV}}}, \end{aligned} \tag{25}$$

where $\widehat{y}_{-i}$ denotes the output of the model built in the ith step of validation process using a data set containing no testing pair $(\mathbf{x}_i, y_i)$ and $\mathrm{MSE}_{\mathrm{CV}}$ is the mean square error.

In order to describe the measure to which the model fits the training data, the root mean square error of training ($\mathrm{RMSE}_{\mathrm{T}}$) is considered. This error is defined as

$$\begin{aligned} \mathrm{MSE}_{\mathrm{T}} &= \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \widehat{y}_i\right)^2, \\ \mathrm{RMSE}_{\mathrm{T}} &= \sqrt{\mathrm{MSE}_{\mathrm{T}}}, \end{aligned} \tag{26}$$

where $\widehat{y}_i$ denotes the output of the model built in the ith step using the full data set and $\mathrm{MSE}_{\mathrm{T}}$ is the mean square error of training.

4. Implementation of the Predictive Models

All the regression models were calculated using *R* language with additional packages [24].

The `lm.ridge` function from "MASS" package [25] was used for calculating OLS regression (where $\lambda = 0$) and ridge regression (where $\lambda > 0$). With the function `enet` included in the package "elastic net" [26], LASSO regression and elastic net regression were calculated. The parameters of the `enet` function are $s \in [0, 1]$ and $\lambda \geq 0$, where s is a fraction of the L_1 norm, whereas λ denotes λ_2 in formula (16). The parameterization of elastic net regression using the pair (λ, s) instead of (λ_1, λ_2) in formula (16) is possible because elastic net regression can be treated as LASSO regression for an augmented data set $(\mathbf{X}^*, \mathbf{y}^*)$ [22]. Assuming that $\lambda = 0$, we get LASSO regression with one parameter s for the original data $(\mathbf{X}, \mathbf{y})$.

All the nonlinear regression models were calculated using the `nls` function coming from the "stats" package [27]. It

calculates the parameters of the model using the nonlinear least squares method. One of the parameters of the `nls` function is a formula that specifies the function $f(\cdot)$ in model (18). To calculate the weights, Gauss-Newton's algorithm was used which was selected by default in the `nls` function. In all the calculations, it was assumed that the initial values of the weights are equal to zero.

For the implementation of artificial neural networks, StatSoft STATISTICA [28] software was used. The learning of MLP networks was implemented using the BFGS (Broyden-Fletcher-Goldfarb-Shanno) algorithm [18]. While calculating the RBF network, the parameters of the basis functions were automatically set by the learning procedure.

The parameters in all models were selected using leave-one-out cross-validation. In the case of regularized regressions, the penalty coefficients were calculated, while, in the case of neural networks, the number of neurons in the hidden layer was calculated. The primary performance criterion of the model was $\mathrm{RMSE}_{\mathrm{CV}}$ error. Cross-validation functions in the STATISTICA program were implemented using Visual Basic language.

5. Results and Discussion

From a coach's point of view, the prediction of results is very important in the process of sport training. A coach using the model, which was constructed earlier, can predict how the training loads will influence the sport outcome. The presented models can be used for predictions based on the proposed monthly training introduced as the sum of the training loads of each type implemented in a given month.

The results of the research are presented in Table 2; the description of the selected regressions will be presented in the next paragraphs. Linear models such as OLS, ridge, and LASSO regressions have been calculated by the authors in work [3]. They will be briefly described here. The nonlinear models implemented using nonlinear regression and artificial neural networks will be discussed in greater detail. All the methods will be compared taking into account the accuracy of the prediction.

5.1. Linear Regressions. The regression model calculated by the OLS method generated the prediction error $\mathrm{RMSE}_{\mathrm{CV}} = 26.90$ s and the training error $\mathrm{RMSE}_{\mathrm{T}} = 22.70$ s (Table 2). In the second column of Table 2, the weights w_0 and w_j are presented.

The search for the ridge regression model is based on finding the parameter λ, for which the model achieves the smallest prediction error. In this paper, ridge regression models for λ changing from 0 to 2 with step of 0.1 were analyzed. Based on the results, it was found that the best ridge model is achieved for $\lambda_{\mathrm{opt}} = 1$. The prediction error $\mathrm{RMSE}_{\mathrm{CV}} = 26.76$ s was smaller than in the OLS model, while the training error $\mathrm{RMSE}_{\mathrm{T}} = 22.82$ s was greater (Table 2). The obtained ridge regression improved the predictive ability of the model. It is seen from Table 2 that as in the case of OLS regression, all weights are nonzero and all the input variables are used in computing the output of the model.

Table 2: Coefficients of linear models and linear part of nonlinear model NLS1 and error results.

Regression	OLS	RIDGE	LASSO, ENET	NLS1
w_0	237.2	325.7	296.6	2005
w_1	45.67	34.67	32.75	41.24
w_2	90.61	74.84	71.91	77.12
w_3	39.70	27.49	24.45	−3.439
w_4	−2.838	2.424		15.45
w_5	−0.9755	−1.770	−1.416	−22.44
w_6	1.072	0.5391		−24.71
w_7	0.7331	0.6805	0.7069	−1.782
w_8	−0.2779	−0.3589	−0.3410	−1.500
w_9	−0.1428	−0.1420	−0.1364	−0.0966
w_{10}	−0.1579	−0.0948	−0.0200	0.7417
w_{11}	0.7472	0.4352	0.0618	0.6933
w_{12}	0.4845	0.3852	0.1793	−0.6726
w_{13}	0.1216	0.1454	0.1183	−0.0936
w_{14}	−0.1510	−0.0270		2.231
w_{15}	−0.5125	−0.3070		0.7349
w_{16}	−0.0601	−0.0571	−0.0652	−0.2685
w_{17}	−0.0153	−0.0071		0.0358
w_{18}	−0.0115	−0.0403	−0.0220	−0.0662
$\mathrm{RMSE}_{\mathrm{CV}}$ [s]	26.90	26.76	26.20	28.83
$\mathrm{RMSE}_{\mathrm{T}}$ [s]	22.70	22.82	22.89	20.21

The LASSO regression model was calculated using the LARS-EN algorithm, in which the penalty is associated with the parameter s changing from 0 to 1 with step of 0.01. It was found that the optimal LASSO regression is calculated for $s_{\mathrm{opt}} = 0.78$. The best LASSO model generates the error $\mathrm{RMSE}_{\mathrm{CV}} = 26.20$ s, which improves the results of OLS and ridge models. However, it should be noted that this model is characterized by the worst data fit with the greatest training error $\mathrm{RMSE}_{\mathrm{T}} = 22.89$ s. The LASSO method is also used for calculating an optimal set of input variables. It can be seen in the fourth column of Table 2 that the LASSO regression eliminated the five input variables (X_4, X_6, X_{14}, X_{15}, and X_{17}), which made the model simpler than for OLS and ridge regression.

The use of elastic net regression model has not improved the value of the prediction error. The best model was obtained for a pair of parameters $s_{\mathrm{opt}} = 0.78$ and $\lambda_{\mathrm{opt}} = 0$. Because the parameter λ is zero, the model is identical to the LASSO regression (fourth column of Table 2).

5.2. Nonlinear Regressions. Nonlinear regression models were obtained using various functions $f(\cdot)$ in formula (18). It was assumed that the function $f(\cdot)$ consists of two components: the linear part, in which the weights are calculated as in OLS regression, and the nonlinear part containing expressions of higher orders in the form of a quadratic function of selected predictors:

$$f(\mathbf{x}_i, \mathbf{w}, \mathbf{v}) = \sum_{j=1}^{p} x_{ij} w_j + \sum_{j=1}^{p} x_{ij}^2 v_j, \tag{27}$$

Table 3: Coefficients of nonlinear part of nonlinear models and error results (all coefficients have to be multiplied by 10^{-2}).

Regression	NLS1	NLS2	NLS3	NLS4
v_5	53.35	−0.3751	−0.3686	−0.6995
v_6	59.43	−1.0454	−1.3869	
v_7	0.1218	0.0003	0.0004	0.0001
v_8	1.880	0.0710	0.0372	−0.0172
v_9	−0.0016	0.0093	0.0093	0.0085
v_{10}	−0.6646	−0.0577	−0.0701	−0.1326
v_{11}	−3.0394	−0.3608	0.0116	0.8915
v_{12}	4.8741	0.3807	0.4170	1.0628
v_{13}	0.4897	−0.2496	−0.2379	−0.1391
v_{14}	−4.7399	−0.1141	−0.1362	
v_{15}	−13.6418	1.3387	0.8183	
v_{16}	0.0335	−0.0015	−0.0003	−0.0004
v_{17}	−0.0033	−0.0006	−0.0006	
v_{18}	0.0054	0.0012	0.0013	−0.0002
$RMSE_{CV}$ [s]	28.83	25.24	25.34	**24.60**
$RMSE_T$ [s]	20.21	22.63	22.74	22.79

where $\mathbf{w} = [w_1, \ldots, w_p]^T$ is the vector of the weights of the linear part and $\mathbf{v} = [v_1, \ldots, v_p]^T$ is the vector of the weights of the nonlinear part. The following cases of nonlinear regression were considered (Table 3), wherein each of the following models does not take into account the squares of qualitative variables X_1, X_2, X_3, and X_4 ($v_1 = v_2 = v_3 = v_4 = 0$):

(i) NLS1: both the weights of the linear part and the weights $v_5, \ldots, v_{18}$ of the nonlinear part are calculated.

(ii) NLS2: the weights of the linear part are constant, and their values come from the OLS regression (the second column of Table 2); the weights $v_5, \ldots, v_{18}$ of the nonlinear part are calculated (the third column of Table 3).

(iii) NLS3: the weights of the linear part are constant, and their values come from the ridge regression (the third column of Table 2); the weights $v_5, \ldots, v_{18}$ of the nonlinear part are calculated (the fourth column of Table 3).

(iv) NLS4: the weights of the linear part are constant, and their values come from the LASSO regression (the fourth column of Table 2); the weights $v_5, v_7, \ldots, v_{13}, v_{16}, v_{18}$ of the nonlinear part are calculated (the fifth column of Table 3).

Based on the results shown in Table 3, the best nonlinear regression model is the NLS4 model, that is, the modified LASSO regression. This model is characterized by the smallest prediction error and the reduced number of predictors.

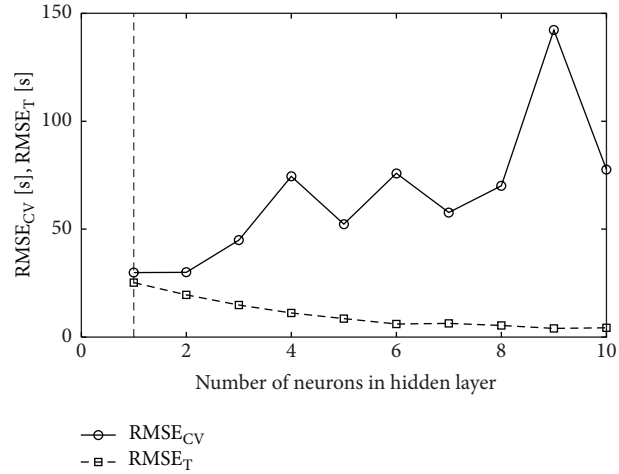


Figure 2: Cross-validation error ($RMSE_{CV}$) and training error ($RMSE_T$) for MLP(tanh) neural network; vertical line drawn for $m = 1$ signifies the number of hidden neurons chosen in cross-validation.

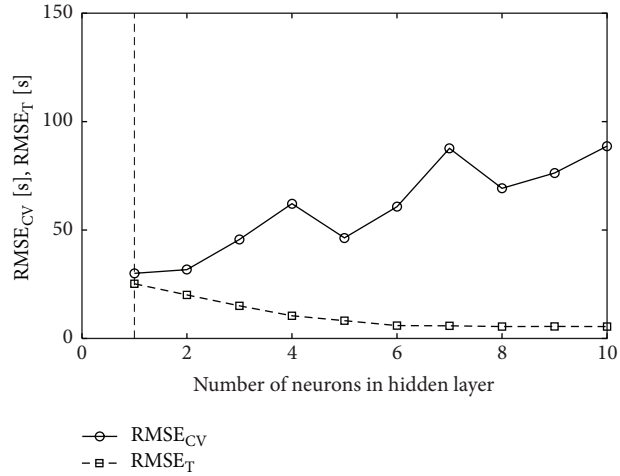


Figure 3: Cross-validation error ($RMSE_{CV}$) and training error ($RMSE_T$) for MLP(exp) neural network; vertical line drawn for $m = 1$ signifies the number of hidden neurons chosen in cross-validation.

5.3. Neural Networks. In order to select the best structure of a neural network, the number of neurons $m \in [1, 10]$ in the hidden layer was analyzed. In Figures 2, 3, and 4, the relationships between cross-validation error and the number of hidden neurons are presented. The smallest cross-validation errors for the MLP(tanh) and MLP(exp) networks were obtained for one hidden neuron (18-1-1 architecture) and they were, respectively, 29.89 s and 30.02 s (Table 4). For the RBF network, the best architecture was the one with four neurons in the hidden layer (18-4-1) and cross-validation error in this case was 55.71 s. Comparing the results, it is seen that the best model is the MLP(tanh) network with the 18-1-1 architecture. However, it is worse than the best regression model NLS4 (Table 3) by more than 5 seconds.

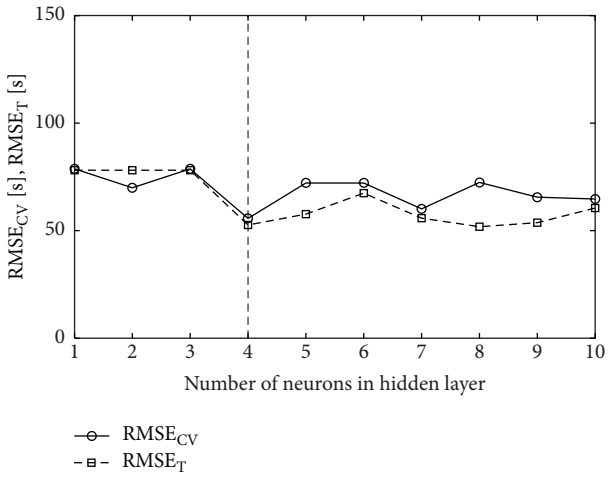


Figure 4: Cross-validation error (RMSE$_{CV}$) and training error (RMSE$_T$) for RBF neural network; vertical line drawn for $m = 4$ signifies the number of hidden neurons chosen in cross-validation.

Table 4: The number of hidden neurons and error results for the best neural nets.

ANN	MLP(tanh)	MLP(exp)	RBF
m	1	1	4
RMSE$_{CV}$ [s]	**29.89**	30.02	55.71
RMSE$_T$ [s]	25.19	25.17	52.63

6. Conclusions

This paper presents linear and nonlinear models used to predict sports results for race walkers. Introducing a monthly training schedule for a selected phase in the annual cycle, a decline in physical performance may be predicted based on the generated results. Thanks to that, it is possible to take into account earlier changes in the scheduled training. The novelty of this research is the use of nonlinear models, including modifications of linear regressions and artificial neural networks, in order to reduce the prediction error generated by linear models. The best model was the nonlinear modification of LASSO regression for which the error was 24.6 seconds. In addition, the method has simplified the structure of the model by eliminating 9 out of 32 predictors. The research hypothesis was confirmed. Comparing with other results is difficult because there is a lack of publications concerning predictive models in race walking.

Experts in the fields of sports theory and training were consulted during the construction of the models in order to maintain the theoretical and practical principles of sport training. The importance of the work is that practitioners (coaches) can use predictive models for planning of training loads in race walking.

Conflict of Interests

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

Acknowledgment

This work has been partially supported by the Polish Ministry of Science and Higher Education under Grant U-649/DS/M.

References

[1] T. O. Bompa and G. Haff, *Periodization: Theory and Methodology of Training*, Human Kinetics, Champaign, Ill, USA, 1999.

[2] A. Maszczyk, A. Zając, and I. Ryguła, "A neural network model approach to athlete selection," *Sports Engineering*, vol. 13, no. 2, pp. 83–93, 2011.

[3] K. Przednowek and K. Wiktorowicz, "Prediction of the result in race walking using regularized regression models," *Journal of Theoretical and Applied Computer Science*, vol. 7, no. 2, pp. 45–58, 2013.

[4] V. Papić, N. Rogulj, and V. Pleština, "Identification of sport talents using a web-oriented expert system with a fuzzy module," *Expert Systems with Applications*, vol. 36, no. 5, pp. 8830–8838, 2009.

[5] R. Roczniok, A. Maszczyk, A. Stanula et al., "Physiological and physical profiles and on-ice performance approach to predict talent in male youth ice hockey players during draft to hockey team," *Isokinetics and Exercise Science*, vol. 21, no. 2, pp. 121–127, 2013.

[6] M. Haghighat, H. Rastegari, N. Nourafza, N. Branch, and I. Esfahan, "A review of data mining techniques for result prediction in sports," *Advances in Computer Science*, vol. 2, no. 5, pp. 7–12, 2013.

[7] K. Przednowek and K. Wiktorowicz, "Neural system of sport result optimization of athletes doing race walking," *Metody Informatyki Stosowanej*, vol. 29, no. 4, pp. 189–200, 2011 (Polish).

[8] A. Drake and R. James, "Prediction of race walking performance via laboratory and field tests," *New Studies in Athletics*, vol. 23, no. 4, pp. 35–41, 2009.

[9] P. Chatterjee, A. K. Banerjee, P. Dasb, and P. Debnath, "A regression equation to predict VO_2 max of young football players of Nepal," *International Journal of Applied Sports Sciences*, vol. 21, no. 2, pp. 113–121, 2009.

[10] B. Ofoghi, J. Zeleznikow, C. MacMahon, and M. Raab, "Data mining in elite sports: a review and a framework," *Measurement in Physical Education and Exercise Science*, vol. 17, no. 3, pp. 171–186, 2013.

[11] E. Mężyk and O. Unold, "Machine learning approach to model sport training," *Computers in Human Behavior*, vol. 27, no. 5, pp. 1499–1506, 2011.

[12] M. Pfeiffer and A. Hohmann, "Applications of neural networks in training science," *Human Movement Science*, vol. 31, no. 2, pp. 344–359, 2012.

[13] I. Ryguła, "Artificial neural networks as a tool of modeling of training loads," in *Proceedings of the 27th Annual International Conference of the Engineering in Medicine and Biology Society (IEEE-EMBS '05)*, pp. 2985–2988, IEEE, September 2005.

[14] A. J. Silva, A. M. Costa, P. M. Oliveira et al., "The use of neural network technology to model swimming performance," *Journal of Sports Science and Medicine*, vol. 6, no. 1, pp. 117–125, 2007.

[15] A. Maszczyk, R. Roczniok, Z. Waśkiewicz et al., "Application of regression and neural models to predict competitive swimming performance," *Perceptual and Motor Skills*, vol. 114, no. 2, pp. 610–626, 2012.

[16] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, New York, NY, USA, 2009.

[17] G. S. Maddala, *Introduction to Econometrics*, Wiley, Chichester, UK, 2001.

[18] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, NY, USA, 2006.

[19] A. E. Hoerl and R. W. Kennard, "Ridge regression: biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[20] R. Tibshirani, "Regression shrinkage and selection via the Lasso," *Journal of the Royal Statistical Society, Series B: Methodological*, vol. 58, no. 1, pp. 267–288, 1996.

[21] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *The Annals of Statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[22] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society—Series B: Statistical Methodology*, vol. 67, no. 2, pp. 301–320, 2005.

[23] S. Arlot and A. Celisse, "A survey of cross-validation procedures for model selection," *Statistics Surveys*, vol. 4, pp. 40–79, 2010.

[24] R Development Core Team, *R: A Language and Environment for Statistical Computing*, R Development Core Team, Vienna, Austria, 2011.

[25] B. Ripley, B. Venables, K. Hornik, A. Gebhardt, and D. Firth, *Package "MASS", Version 7.3–20*, CRAN, 2012.

[26] H. Zou and T. Hastie, *Package "elasticnet", version 1.1*, CRAN, 2012.

[27] R Development Core Team and Contributors Worldwide, *The R "Stats" Package*, R Development Core Team, Vienna, Austria, 2011.

[28] StatSoft, *Statistica (Data Analysis Software System), Version 10*, StatSoft, 2011.