Novel Particle Swarm Optimization Algorithms with Applications to Healthcare Data Analysis

Weibo Liu

Department of Computer Science
Brunel University London

This dissertation is submitted for the degree of
Doctor of Philosophy

February 2020

I would like to dedicate this thesis to my family.

Declaration

I, Weibo Liu, hereby declare that this thesis and the works presented in it are entirely my own. Some of the works have been previously published in journal or conference papers, and this has been mentioned in the thesis. Where I have consulted the work of others, this is always clearly stated.

Weibo Liu

February 2020

Acknowledgements

At the beginning, I would like to take this opportunity to express my deepest gratitude to my principle supervisor Prof. Zidong Wang and my second supervisor Prof. Xiaohui Liu for supporting and guiding my Ph.D. study.

It is my great honor to have Prof. Wang as my Ph.D. supervisor. He is a wonderful supervisor, and I really appreciate all his contributions, efforts, ideas and time to make my Ph.D. experience productive and enjoyable. Not only his insightful comments, professional guidance but also his invaluable encouragement and constant support taught me what high-quality research is and how to be a mature researcher. Under his guidance, I learnt to be a responsible person, which will indeed benefit my life. I would also like to take this opportunity to express my sincere thanks to Prof. Liu for all his encouragement and kind help in the past four years. I have always appreciated his support and help to me whenever I face difficulties.

I am truly grateful to Prof. Yurong Liu in China, who encouraged me to apply for the Ph.D. at Brunel University London and has given me continued support over the past years. My sincere thanks also go to my house mates: Prof. Yuan Yuan, Dr. Fan Wang, Dr. Hongwei Chen, Dr. Licheng Wang, Dr. Shuai Liu, Ms. Di Zhao and Mr. Yuxuan Shen for building an enjoyable place to live in. It is my honor to have you with me in these years.

I would also like to take this opportunity to express my gratitude to the following people for helpful discussions, suggestions, comments, and support of my research during my Ph.D. stage: Prof. Yurong Liu, Prof. Jun Hu, Dr. Lei Zou, Dr. Liang Hu, Prof. Lifeng Ma, Dr. Hang Geng, Prof. Yingjie Tian, Dr. Qinyuan Liu, Mr. Wenshuo Li, Dr. Nianyin Zeng, Ms. Hanjing Cheng, Dr. Yonggang Chen, Dr. Wenying Xu, Dr. Bing

Li, Dr. Baoye Song, Dr. Yan Song, Prof. Bo Shen, Prof. Xin Luo, Dr. Lei Ma, Mr. Bo Tian, Dr. Hua Yang, Dr. Lulu Tian, Ms. Chen Gao, and Mr. Weihao Song.

I would like to thank all my lab mates at the Intelligent Data Analysis Centre for the pleasant working environment: Dr. Liang Hu, Mr. Wenbin Yue, Ms. Yani Xue, Dr. Miqing Li, Dr. Chuang Wang, Dr. Mahir Arzoky, Dr. Zairul Mazwan, Dr. Neda Trifonova, Dr. Izaz Rahman, Dr. Mohsina Ferdous, Dr. Khalid Eltayef, Mr. Bashir Dodo, Dr. Navid Dorudian, and Ms. Shakirat Adesola. I would also like to thank the Department of Computer Science, Brunel University London for supporting my Ph.D. research.

The last but not the least, I would like to express my deepest thanks and gratitude to my family. My parents provided me with unconditional support and encouragement all the time, which motivates me to move forward. They never complained and always dedicated their endless love and happiness to me. Thank you so much for all your love and care!

Abstract

Optimization problem is a fundamental research topic which has been receiving increasing interest according to its application potential in almost all real-world systems including engineering systems, large-scaled complex networks, healthcare management systems and so on. A large number of heuristic algorithms have been developed with the purpose of effectively solving the optimization problems during the past few decades. Served as a powerful family of heuristic algorithms, the particle swarm optimization (PSO) algorithm has been successfully employed in a variety of practical applications in dealing with optimization problems. The PSO algorithm has exhibited more competitive performance than many popular evolutionary computation approaches because of its easy implementation, fast convergence and comprehensive ability of converging to a satisfactory solution. Nevertheless, there is still much room to improve the PSO algorithm in terms of both the convergence rate and the population diversity.

To summarize, there are three challenging problems in developing new variant PSO algorithms with hope to further improve the convergence rate of the PSO algorithm and maintain the population diversity: 1) how to adjust the control parameters of the PSO algorithm; 2) how to achieve the balance between the local search and the global search during the evolution process; and 3) how to guarantee the search ability of the particles and avoid premature convergence.

In this thesis, we address the above mentioned challenging problems and aim to design effective variant PSO algorithms with applications in intelligent data analysis. It should be pointed out that all the developed PSO algorithms in this thesis have been evaluated by comparing with some currently popular variant PSO algorithms.

- With the aim to improve the convergence rate of the optimizer, an adaptive weighting PSO algorithm is put forward where a sigmoid-function-based weighting

strategy is introduced to adjust the acceleration coefficients. With this weighting strategy, the distances from the particle to the global best position and from the particle to its personal best position are both taken into consideration, thereby having the distinguishing feature of enhancing the convergence rate.

- As with other evolutionary computation approaches, the modification of parameters is an efficient method for improving the search ability of the algorithm. We present a randomised PSO algorithm where Gaussian white noise with adjustable intensity is utilized to randomly perturb the acceleration coefficients in order to explore and exploit the problem space thoroughly.

- To further develop a novel PSO algorithm with promising search ability, we propose a randomly occurring distributedly delayed particle swarm optimization (RODDPSO) algorithm which demonstrates competitive performance in seeking the optimal solution. The randomly occurring distributed time delays not only contribute to a thorough exploration of the search space but also achieve a proper balance between the local exploitation and the global exploration.

- To fully investigate the application potential of the developed PSO algorithms, we apply the RODDPSO algorithm to intelligent data analysis (including data clustering and classification problems). We optimize the initial cluster centroids of the K-means clustering algorithm and the hyperparameters of the deep neural network by using the RODDPSO algorithm. The developed PRODDPSO-based algorithms are successfully employed in patients' triage categorization and patient attendance disposal problems with satisfactory performance.

Table of contents

List of figures **xi**

List of tables **xiii**

Nomenclature **xiv**

1 Introduction **1**

1.1 Motivation 1

1.2 Contribution 3

1.3 Publication 6

1.4 Thesis Structure 7

2 Background **9**

2.1 Particle Swarm Optimization 9

2.2 Development of Particle Swarm Optimization Algorithms 11

2.3 Applications of Particle Swarm Optimization in Intelligent Data Analysis 16

2.3.1 Data Clustering 16

2.3.2 Deep Learning 17

3 A Novel Sigmoid-Function-Based Adaptive Weighted Particle Swarm Optimizer **21**

3.1 Motivation 21

3.2 A Novel AWPSO Algorithm 23

3.2.1 Adaptive Weighting Strategy 24

3.2.2 Selection of Adaptive Weighting Updating Function 25

3.2.3 Framework of the AWPSO Algorithm 27
3.3 Results and Discussions 29
3.3.1 Benchmark Functions 29
3.3.2 Experiment Results 31
3.4 Conclusion 44

4 A Novel Randomised Particle Swarm Optimizer 45
4.1 Motivation 45
4.2 A New RPSO Algorithm 47
4.2.1 Framework of the RPSO Algorithm 48
4.3 Results and Discussion 50
4.3.1 Test Functions 50
4.3.2 Experimental Studies of the RPSO Algorithm 53
4.4 Conclusion 59

5 A Randomly Occurring Distributedly Delayed Particle Swarm Optimizer 61
5.1 Motivation 61
5.2 A Novel RODDPSO Algorithm 63
5.2.1 Framework of the RODDPSO Algorithm 63
5.2.2 Evolutionary State 64
5.2.3 Velocity Updating Strategy Based on Randomly Occurring Distributed Time-delay 65
5.3 Results and Discussions 66
5.3.1 Selection of Benchmark Functions 66
5.3.2 Experiment Results of the RODDPSO Algorithm 68
5.4 Conclusion 72

6 Applications of Particle Swarm Optimization Algorithms in Intelligent Data Analysis 79
6.1 Motivation 79
6.2 A&E Data 82

6.2.1 Data Description 82
6.3 Patient Clustering in A&E Data 83
6.3.1 A RODDPSO-Based Clustering Algorithm 84
6.3.2 Results and Discussions 85
6.4 Patient Classification in A&E Data 89
6.4.1 Deep Belief Network and Restricted Boltzmann Machine 89
6.4.2 A Novel RODDPSO-Based DBN 93
6.4.3 Experiment Results and Discussions 95
6.5 Conclusion 100

7 Conclusions and Future Research **106**
7.1 Summarization 106
7.2 Future Work 108

References **111**

List of figures

2.1 Flowchart of the traditional PSO algorithm 12

3.1 Flowchart of the AWPSO algorithm 28
3.2 Optimization performance for Sphere function $f_1(x)$ 33
3.3 Optimization performance for Rosenbrock function $f_2(x)$ 33
3.4 Optimization performance for Rastrigin function $f_3(x)$ 34
3.5 Optimization performance for Schwefel 2.22 function $f_4(x)$ 34
3.6 Optimization performance for Penalized 1 function $f_5(x)$ 35
3.7 Optimization performance for Step function $f_6(x)$ 35
3.8 Optimization performance for Schwefel function $f_7(x)$ 36
3.9 Optimization performance for Penalized 2 function $f_8(x)$ 36
3.10 Diversity measure for Sphere function $f_1(x)$ 37
3.11 Diversity measure for Rosenbrock function $f_2(x)$ 38
3.12 Diversity measure for Rastrigin function $f_3(x)$ 38
3.13 Diversity measure for Schwefel 2.22 function $f_4(x)$ 39
3.14 Diversity measure for Penalized 1 function $f_5(x)$ 39
3.15 Diversity measure for Step function $f_6(x)$ 40
3.16 Diversity measure for Schwefel function $f_7(x)$ 40
3.17 Diversity measure for Penalized 2 function $f_8(x)$ 41
3.18 Convergence plot of PSO algorithms 43

4.1 Flowchart of the RPSO algorithm 49
4.2 Algorithm Convergence Characteristics for Sphere function $f_1(x)$ . . . 53
4.3 Algorithm Convergence Characteristics for Ackley function $f_2(x)$ . . . 54

4.4 Algorithm Convergence Characteristics for Rastrigin function $f_3(x)$ . . 54
4.5 Algorithm Convergence Characteristics for Schwefel 2.22 function $f_4(x)$ 55
4.6 Algorithm Convergence Characteristics for Griewank function $f_5(x)$ . . 55
4.7 Algorithm Convergence Characteristics for Penalized 1 function $f_6(x)$. 56
4.8 Algorithm Convergence Characteristics for Step function $f_7(x)$ 56
4.9 Algorithm Convergence Characteristics for Penalized 2 function $f_8(x)$. 57

5.1 The flowchart of the RODDPSO algorithm 74
5.2 Performance test for Sphere function $f_1(x)$ 75
5.3 Performance test for Rosenbrock function $f_2(x)$ 75
5.4 Performance test for Ackley function $f_3(x)$ 76
5.5 Performance test for Rastrigin function $f_4(x)$ 76
5.6 Performance test for Schwefel 2.22 function $f_5(x)$ 77
5.7 Performance test for Schwefel 1.2 function $f_6(x)$ 77
5.8 Performance test for Griewank function $f_7(x)$ 78
5.9 Performance test for Step function $f_8(x)$ 78

6.1 Silhouette coefficient of K-means clustering algorithm 87
6.2 Silhouette coefficient of Fuzzy C-means clustering algorithm 88
6.3 Silhouette coefficient of RODDPSO-based clustering algorithm 88
6.4 The architecture of an RBM 90
6.5 The architecture of a DBN 101
6.6 The diagram of output classes (acquired from real labels). The detailed descriptions of each label and class are given in Table 6.2 102
6.7 Full-batch training mean squared error results of the DBN 103
6.8 Full-batch training mean squared error results of the DBN 103
6.9 Full-batch training mean squared error results of the DBN 104
6.10 Accuracy of the KNN algorithm when k is selected from 1 to 100 . . . 104
6.11 Full-batch training mean squared error results of three DBNS (RODDPSO-based DBN, Penalized DBN, and standard DBN 105

List of tables

3.1 Configuration of benchmark functions 31
3.2 Algorithm evaluation on eight benchmark functions 32
3.3 Convergence performance evaluation by using the ranking method . . 43

4.1 Test function configuration 52
4.2 Statistical Results of PSO Algorithms 58
4.3 Convergence performance evaluation by using the ranking method . . 59

5.1 Velocity Updating Strategy for Distributed Time-delayed Information . 66
5.2 Configuration of benchmark functions 67
5.3 Performance evaluation of RODDPSO algorithm with different N . . . 69
5.4 Comparisons of various PSO algorithms on eight optimization benchmark functions 70
5.5 Convergence performance evaluation by using the ranking method . . 71
5.6 Comparisons of various PSO algorithms in 50-dimensional search space 72

6.1 Patients Attendance at the Emergency Departments 82
6.2 Patient Attendance Disposal 96
6.3 Modified class for patient attendance disposal 97
6.4 Configuration of the Standard DBN at the pre-training stage 98

Nomenclature

Acronyms / Abbreviations

A&E	accident & emergency
AI	artificial intelligence
ANN	artificial neural network
DBN	deep belief network
DNN	deep neural network
EA	evolutionary algorithm
EC	evolutionary computation
EP	evolution programming
GA	genetic algorithm
GWN	Gaussian white noise
IDA	intelligent data analysis
KNN	k-nearest neighbours
ML	machine learning
MSE	mean square error
NHS	national health service

NLP natural language processing

PSO particle swarm optimization

RBM restricted Boltzmann machine

Chapter 1

Introduction

1.1 Motivation

In recent years, the study of optimization techniques has attracted enormous research interest from a variety of research communities such as computer science, information science and communication. It is well known that evolutionary computation (EC) serves as a powerful family of algorithms that can be effectively used to solve global optimization problems by using stochastic or metaheuristic searching strategies. Some important EC approaches, which include evolutionary programming, evolutionary strategies, genetic algorithms and generic programming, are motivated by biological evolution and have been successfully applied to various research fields (e.g. artificial intelligence, signal processing, and telecommunication), see [40, 89]. Among others, the particle swarm optimization (PSO) algorithm proposed in [46] has been inspired by simulating social behaviors such as bird flocking and fish schooling. The PSO algorithm has been recognized as a particularly attractive EC algorithm that has found a wide range of applications in dealing with optimization problems.

PSO is a population-based heuristic algorithm that starts with random initialization of a group of individuals in the search space [129, 27, 35, 46]. These individuals are defined as particles, where each particle represents a candidate solution and a group of particles is referred to as a swarm. Moreover, the particles are trying to fly to their own best position as well as the global best position among the whole swarm

at each step according to the social behaviors in the swarm. The PSO algorithm has received much research attention owing to its easy implementation and competitiveness in finding a relatively satisfactory solution with a reasonable convergence rate, see e.g., [154, 47, 93, 48, 192, 149, 188, 140].

As with almost all EC algorithms, the PSO algorithms suffer from the problem of trapping local optima especially in high-dimensional optimization processes. Consequently, it is of practical significance to develop advanced approaches to further improve the search ability of the PSO algorithms in terms of both the convergence and the diversity. In the past few years, a variety of improved PSO algorithms have been put forward to enhance the search ability of the PSO algorithm and reduce the possibility of getting trapped in the local optima, see e.g., [192, 149, 140, 188]. For example, an adaptive PSO (APSO) algorithm has been proposed in [192] by developing a systematic parameter adaptation scheme based on the evolutionary factor to automatically control the parameters such as the inertia weight and acceleration coefficients. A switching PSO (SPSO) algorithm has been developed in [149] where the velocity model updates according to a Markov chain, thereby exploring the search space more thoroughly than the APSO algorithm. Furthermore, a switching delayed PSO (SDPSO) algorithm has been introduced in [188] where the delayed information (containing previous personal best and global best particles) has been used to further enhance the searching capability. Moreover, a multimodal delayed PSO (MDPSO) algorithm has been proposed in [140] where the multimodal time-delays (added in the velocity updating model) have helped reduce the possibility of being trapped in the local optimum and expand the search space. Nevertheless, there is still much room to further improve the performance of the aforementioned algorithms especially for high-dimensional optimization problems with a large number of local optima.

It should also be pointed out that although some popular PSO algorithms have exhibited competitive performance on searching for the global optimum and increasing the possibility of avoiding the local optima, the enhancement of the search performance of PSO algorithms is often at the expense of sacrificing the convergence rate, which is certainly undesirable [25, 30, 23]. As such, it is of practical significance to develop a

new PSO algorithm that is capable of finding the globally optimal solution yet with a *satisfactory* convergence rate.

Motivated by above discussions, the main challenging problems of improving the PSO algorithms can be summarized into the following aspects: 1) how to improve the convergence rate of the PSO algorithms? 2) how to develop new PSO algorithms that can achieve an adequate balance between the local exploitation and global exploration during the evolution process? 3) how to enhance the search capability of the optimizer to avoid premature convergence?

The main purpose of this thesis is to launch a major study on developing a number of innovative algorithms with hope to overcome the aforementioned difficulties in PSO algorithms. In Chapter 3, we develop an adaptive weighting PSO (AWPSO) algorithm in order to improve the convergence rate of the PSO algorithm. In Chapter 4, a randomised PSO (RPSO) algorithm is put forward to randomly perturb the acceleration coefficients in order for the problem space to be explored more thoroughly. We endeavor to propose a randomly occurring distributedly delayed PSO (RODDPSO) in Chapter 5. The applications of the developed PSO algorithm is further addressed in Chapter 6 where the RODDPSO algorithm is successfully applied to solve the parameter selection problems in intelligent data analysis (where in this thesis the clustering and classification problems are addressed) on A&E department.

1.2 Contribution

The main contributions of this thesis are outlined as follows.

- We develop an AWPSO algorithm where a sigmoid-function-based parameter selection strategy is designed in order to improve the convergence rate of the PSO algorithm. In the AWPSO algorithm, we endeavor to make full use of the distances from each particle to its personal best position and the global best position at each iteration. The sigmoid-function-based parameter selection strategy adaptively adjusts the acceleration coefficients based on the outputs of a sigmoid function with the computed distances as the inputs. Comparing with the time-varying parameter updating strategy in PSO algorithms, the sigmoid-function-based

parameter updating mechanism not only adaptively controls the acceleration coefficients, which helps guarantee the population diversity, but also contributes to a relatively fast exploration of the search space by forcing the particles to search around the personal best position and the global best position as fast as necessary. A series of experiments are carried out to comprehensively validate the effectiveness of the developed AWPSO algorithm via some well-known benchmark functions. Experiment results show that the AWPSO algorithm outperforms four currently popular variant PSO algorithms.

- We propose a RPSO algorithm to further improve the population diversity and alleviate the premature convergence problem. The Gaussian white noise (GWN) with moderate intensity is applied to randomly perturb the acceleration coefficients of the RPSO algorithm, which contributes to a more thorough exploration of the search space. The utilization of the GWNs in the acceleration coefficients could alter the system dynamics (at each iteration), which leads to a more thoroughly exploited and explored search space. Additionally, the particles are entitled to exhibit more complicated dynamical behaviors than the standard PSO algorithm by utilizing the GWNs, which would enhance the capability of the particles getting rid of the local optima and improve the population diversity. The RPSO algorithm is verified on a series of benchmark functions. Experiment results demonstrate that the RPSO algorithms outperforms some existing PSO algorithms via some widely used optimization benchmark functions.

- We put forward a RODDPSO algorithm where the randomly occurring distributed time delays are introduced in the velocity updating model with hope to improve the search ability of the optimizer in terms of the convergence and the population diversity. The utilization of the time-delay terms (composed of both personal and global best particles in the velocity updating model) could not only make full use of the historical information during the evolution process but also exhibit a more complicated dynamical behavior, which leads to less possibility of being trapped in the local optima. Furthermore, the introduced distributed time-delays occur randomly with reasonably small probability, which plays an adequate

tradeoff between the convergence and the diversity. Eight well-known benchmark functions are employed to evaluate the performance of the proposed RODDPSO algorithm. Experiment results demonstrate the superiority of the RODDPSO algorithm over several currently popular PSO algorithms.

- We propose a new clustering algorithm to combine the RODDPSO algorithm with the well-known K-means clustering algorithm. The proposed RODDPSO-based clustering algorithm is not dependent on the initial states of the cluster centroids, thereby facilitating a better cluster partition. The developed RODDPSO-based clustering algorithm is applied to solve the triage categorization problem in the accident & emergency (A&E) departments. The clustering performance is evaluated by using the Silhouette coefficient. Experiment results show that the RODDPSO-based clustering algorithm is superior to the traditional clustering algorithms on the A&E data. With an appropriate triage category (resulting in improved patient routing), the patients' waiting time within A&E departments could be much decreased and patients with serious injury or illness can then be treated with specific care. As such, the efficiency of both human and non-human resource management in A&E departments could be improved.

- We propose a hyperparameter selection mechanism and apply it to the deep belief network (DBN) where the hyperparameters (e.g., learning rate, momentum, weight decay) are tuned via the RODDPSO algorithm. Instead of randomly choosing the hyperparameters or selecting the hyperparameters based on empirical experience, the hyperparameters of the DBN are determined according to optimization results by employing the RODDPSO algorithm. The hyperparameters in the pre-training process and the fine-tuning process are optimized, which results in a better classification performance than that of the standard DBN. The proposed RODDPSO-based DBN is successfully applied to deal with patient attendance disposal problem in an A&E department. Experiment results demonstrates the effectiveness of the developed approach on analyzing A&E data. With the patient discharge categories (obtained by the output of the network), the patients could be efficiently assigned to a hospital bed and transferred to

other clinics. The resource management of the A&E departments could also be improved.

1.3 Publication

The following papers report the research work resulting from this thesis.

- **W. Liu**, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, A survey of deep neural network architectures and their applications, *Neurocomputing*, vol. 234, pp. 11–26, 2017. (Resulting from Chapter 2)
- **W. Liu**, Z. Wang, Y. Yuan, N. Zeng, and X. Liu, A novel sigmoid-function-based adaptive weighted particle swarm optimizer, *IEEE Transactions on Cybernetics*, 2019, in press, DOI: 10.1109/TCYB.2019.2925015. (Resulting from Chapter 3)
- **W. Liu**, Z. Wang, X. Liu, N. Zeng, and D. Bell, A novel particle swarm optimization approach for patient clustering from emergency departments, *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 4, pp. 632–644, 2019. (Resulting from Chapters 5, 6)
- **W. Liu**, Z. Wang, N. Zeng, Y. Yuan, and X. Liu, A novel randomised particle swarm optimizer, under review (submitted to *IEEE Transactions on Cybernetics*). (Resulting from Chapter 4)
- **W. Liu**, Z. Wang, W. Yue, and D. Bell, A clustering approach to triage categorization in A&E departments, In: *Proceedings of the 23rd International Conference on Automation and Computing*, Huddersfield, UK, Sept. 2017. (Resulting from Chapter 6)
- N. Zeng, H. Qiu, Z. Wang, **W. Liu**, H. Zhang, and Y. Li, A new switching-delayed-PSO-based optimized SVM algorithm for diagnosis of Alzheimer's disease, *Neurocomputing*, vol. 320, pp. 195–202, 2018.
- N. Zeng, H. Zhang, B. Song, **W. Liu**, Y. Li, and A. M. Dobaie, Facial expression recognition via learning deep sparse autoencoders, *Neurocomputing*, vol. 273, pp. 643–649, 2018.

1.4 Thesis Structure

To summarize, this thesis is organised into 7 chapters including the current chapter (an introduction of this thesis). The contents of the rest chapters are summarized in the following manner.

In Chapter 2, we provide the necessary background information of the knowledge related to this thesis. First of all, we briefly introduce the basic knowledge of the PSO algorithm, and the applications of the PSO algorithm are presented with details. Then, we review the development of the variant PSO algorithms in the following three aspects: 1) adjust the control parameters; 2) design new typology structures; and 3) develop hybrid PSO algorithms by combining other search techniques. Next, the PSO-based clustering algorithms are introduced where traditional clustering algorithms are also investigated. Finally, the utilization of PSO algorithms in optimising the hyperparameters in a deep learning architecture is presented.

Motivated by previous work, in Chapter 3, a novel particle swarm optimization algorithm is put forward where a sigmoid-function-based weighting strategy is developed to adaptively adjust the acceleration coefficients. The newly proposed adaptive weighting strategy takes into account both the distances from the particle to the global best position and from the particle to its personal best position, thereby having the distinguishing feature of enhancing the convergence rate. Inspired by the activation function of neural networks, the new strategy is employed to update the acceleration coefficients by using the sigmoid function. The search capability of the developed AWPSO algorithm is comprehensively evaluated via eight well-known benchmark functions including both the unimodal and multimodal cases. Simulation results demonstrate that the designed AWPSO algorithm substantially improves the convergence rate of the particle swarm optimizer and also outperforms some currently popular PSO algorithms.

Chapter 4 discusses a novel RPSO algorithm where the Gaussian white noise with adjustable intensity is utilized to randomly perturb the acceleration coefficients in order for the problem space to be explored more thoroughly. With this new strategy, the RPSO algorithm not only maintains the population diversity but also enhances the possibility of escaping the local optima trap. Experiment results demonstrate that

the proposed RPSO algorithm outperforms some existing popular variants of PSO algorithms on a series of widely used optimization benchmark functions.

In Chapter 5, a novel RODDPSO algorithm is proposed where the evolutionary state is determined by evaluating the evolutionary factor in each iteration, based on which the velocity updating model switches from one mode to another. With the purpose of reducing the possibility of being trapped in the local optima and expanding the search space, randomly occurring time-delays that reflect the history of previous personal best and global best particles are introduced in the velocity updating model in a distributed manner. Eight well-known benchmark functions are employed to evaluate the proposed RODDPSO algorithm, which is shown via extensive comparisons to outperform some currently popular PSO algorithms.

In Chapter 6, the applications of the developed RODDPSO algorithm is discussed: 1) the developed RODDPSO algorithm is employed to design an improved clustering algorithm. The novel RODDPSO-based clustering algorithm combines the RODDPSO algorithm with the traditional K-means clustering algorithm. The procedure of the algorithm is described in a flowchart. To further illustrate the application potential, the developed RODDPSO-based clustering algorithm is successfully exploited in the patient clustering problem for data analysis with respect to a local A&E department in West London. Experiment results demonstrate that the RODDPSO-based clustering method is superior over two other well-known clustering algorithms which are the K-means clustering algorithm and the fuzzy C-means clustering algorithm. 2) We apply the RODDPSO algorithm to optimize the hyperparameters of the popular deep learning technique, the DBN. A novel RODDPSO-based DBN is put forward with hope to improve the classification performance of the traditional DBN. The basic DBN is presented where the parameter updating mechanism is given with detailed information. Then the learning algorithm of the developed RODDPSO-based DBN is illustrated and the proposed model is employed on analyzing the patient attendance disposal problem. Experiment results demonstrate the superiority of the proposed model over the traditional DBN and the penalized DBN (with penalty and momentum).

In Chapter 7, the work of this thesis is concluded and some relevant future research directions are presented.

Chapter 2

Background

In this chapter, we aim to review the development of the PSO algorithm and its applications. First, the background of the PSO algorithm is introduced in Section 2.1. Then, the development of the variant PSO algorithms are further addressed in Section 2.2. Due to the rapid growth of the literature, it is impossible for us to review all recently proposed PSO algorithms that are related to our study. In this case, we focus on three main types of variant PSO algorithms which include: 1) PSO algorithms with new parameter updating mechanisms; 2) PSO algorithms with new learning strategies; and 3) PSO algorithms hybridized with other EC algorithms. Finally, the practical applications of PSO algorithms in intelligent data analysis (IDA) are further summarized in Section 2.3.

2.1 Particle Swarm Optimization

Owing to their practical application insights, optimization problems have drawn considerable research attention from both industrial and academic societies. The past few years have witnessed a rapid development of optimization techniques developed by various research communities including computer science, mathematics, control engineering and signal processing. In particular, as a powerful group of optimization techniques, the EC approaches have proven to be highly efficient in solving global optimization problems with great application potentials, and have therefore attracted tremendous research interest [33, 172]. Motivated by the biological evolution, many

well-known EC approaches (e.g. the PSO algorithm, EP, and the GA) have been successfully employed to a variety of real-world applications in the research areas of artificial intelligence, signal processing and system science [135, 136, 46, 48, 153, 190, 192, 173, 49, 134]. Compared with other popular EC algorithms (such as the GA, differential evolution, and simulate annealing), the PSO algorithm exhibits competitive or even superior performance and is thus recognized as an excellent candidate algorithm due mainly to its technical merits of easy implementation and fast convergence towards the optimal solution [92, 82].

Motivated by the mimics of the social interactions (e.g. fish schooling or birds flocking), the PSO algorithm aims to explore the search space by adjusting the velocity and position of particles according to the swarm intelligence. In fact, the PSO algorithm is capable of discovering the optimal solution both effectively and efficiently, and has been regarded as a rather powerful optimization technique [171, 101, 22, 159, 21, 113]. So far, the PSO algorithm has been successfully applied to solve the optimization problems in a wide range of real-world systems such as power systems [35, 7], genetic regulatory networks [149], medical systems [188, 190] and path planning systems [140]. In the execution of a PSO algorithm, by cooperating and competing with other individual particles, each particle is encouraged to learn from experience of itself and competitors to seek the globally optimal solution through the entire search space. During the evolution process at each iteration, each individual particle is guided by its historical personal best position and the global best position discovered by the entire yet dynamical swarm.

Inspired by a metaphor of social interaction, the PSO algorithm is developed to simulate the social behavior of fish schooling or birds flocking, where each particle represents a candidate solution of the research problem. Note that all the particles move at a certain speed in a D-dimensional search space. The velocity and position of the ith particle at the kth iteration are denoted by two vectors, which are the velocity vector $v_i(k) = (v_{i1}(k), v_{i2}(k), \cdots, v_{iD}(k))$ and the position vector $x_i(k) = (x_{i1}(k), x_{i2}(k), \cdots, x_{iD}(k))$, respectively. According to the swarm intelligence, the position of each particle is automatically updated in the direction of the global optimum, where one is the personal best position found by itself (pbest) denoted by $p_i =$

$(p_{i1}, p_{i2}, \cdots, p_{iD})$, and the other one is the global best position throughout the whole swarm (gbest) represented by $p_g = (p_{g1}, p_{g2}, \cdots, p_{gD})$. The velocity and position of the ith particle at the $(k+1)$th iteration are updated as follows:

$$\begin{aligned} v_i(k+1) &= wv_i(k) + c_1 r_1(p_i(k) - x_i(k)) + c_2 r_2(p_g(k) - x_i(k)), \\ x_i(k+1) &= x_i(k) + v_i(k+1), \end{aligned} \tag{2.1}$$

where k is the current iteration number; w is the inertia weight; c_1 and c_2 are the acceleration coefficients called as cognitive and social parameters, respectively; and r_1 and r_2 are two random numbers which are uniformly distributed over the interval $[0, 1]$. The inertia weight as well as acceleration coefficients, which serve as another two control parameters, are vitally important in the velocity updating model of the PSO algorithm and have been extensively investigated in recent years for better accuracy and faster convergence rate [30, 129, 145, 194, 195, 193].

The flowchart of the traditional PSO algorithm is depicted in Fig. 2.1.

2.2 Development of Particle Swarm Optimization Algorithms

Population-based EC approaches are known to have the problems of easily being trapped in the local optima especially in the handling of large-scale optimization problems. The PSO algorithm is not an exception where the individual particle in executing a PSO algorithm may easily be trapped in the local optima, and this leads to the so-called premature convergence. Under this circumstance, despite the ongoing effort, it is still vitally important to further develop advanced algorithms in order to enhance global search capability of the PSO algorithms [160, 149, 140].

Up to now, a great deal of research attention has been paid to the improvement of the search capability of the existing PSO algorithms by developing advanced PSO variants so as to alleviate the phenomenon of premature convergence, see in [135, 136, 23, 25, 192, 97, 31, 188, 181, 197, 111, 19]. To be specific, three types of PSO variants have been introduced by: 1) putting forward novel strategies to adjust the control

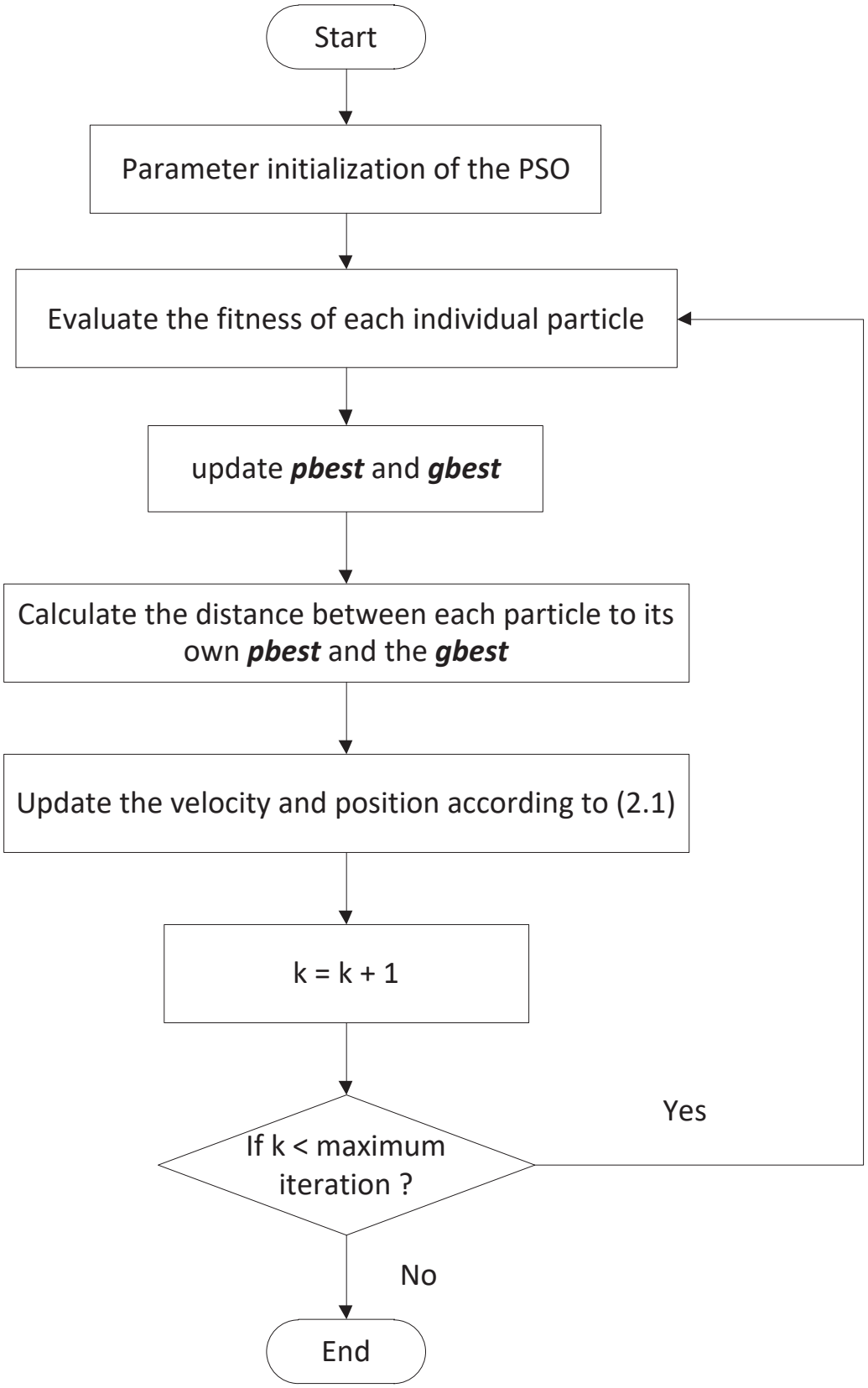

Fig. 2.1 Flowchart of the traditional PSO algorithm

parameters; 2) designing new updating topological structures and 3) hybridizing with other EC algorithms.

Modify Parameters in Particle Swarm Optimization

In the past few decades, the problem of improving traditional PSO algorithms has been attracting particular research attention. Various modified PSO algorithms have been proposed to enhance the search ability of the traditional PSO algorithm. It is well known that, as a control parameter, the balance between global and local searches throughout the searching process plays a vital role in successfully finding the optimal solution [98, 145, 181, 71]. Up to now, some PSO variants have been focused on the modification of the three control parameters in PSO algorithms: the inertia weight, the cognitive acceleration coefficient, and the social acceleration coefficient.

In a PSO algorithm, the inertia weight is normally utilized to balance the global search and the local search, where a larger value of the inertia weight contributes to a better global exploration, and a smaller value encourages a more thorough local exploitation [136]. In [135, 136], a linear-decreasing-inertia-weight-based PSO (PSO-LDIW) algorithm has been proposed where the inertia weight is updated in a time-varying manner. The updating function of the inertia weight in the PSO-LDIW is expressed as follows:

$$w = w_{\max} - (w_{\max} - w_{\min}) \times \frac{\text{iter}}{\text{maxiter}}, \tag{2.2}$$

where $w_{\max}$ and $w_{\min}$ represent the maximum and minimum value of the inertia weight, respectively; *iter* denotes the number of current iteration, and *maxiter* represents the maximum iteration number. Normally, a larger inertia weight will benefit the global exploration, and a smaller inertia weight will contribute to the local exploitation [136]. The PSO-LDIW algorithm has satisfactory performance in many applications. However, for the PSO-LDIW algorithm, once the inertia weight decreases, the search ability of the swarm would be affected and may not explore the search space thoroughly [35].

For the purpose of efficiently controlling the local and global searches, the time-varying-acceleration-coefficient-based PSO (PSO-TVAC) algorithm has been introduced in [129]. The cognitive acceleration coefficient c_1 is linearly decreased, and the social acceleration coefficient c_2 is linearly increased, which are shown as follows:

$$c_1 = (c_{1f} - c_{1i}) \times \frac{\text{maxiter} - \text{iter}}{\text{maxiter}} + c_{1i}, \tag{2.3}$$

$$c_2 = (c_{2f} - c_{2i}) \times \frac{\text{maxiter} - \text{iter}}{\text{maxiter}} + c_{2i}, \tag{2.4}$$

where c_{1i} and c_{2i} represent the initial values of the acceleration coefficients. c_{1f} and c_{2f} denote the final value of the cognitive acceleration coefficient c_1 and the social acceleration coefficient c_2, respectively. It should be mentioned that the parameters $c_{1i} = 0.5$, $c_{1f} = 2.5$, $c_{2i} = 2.5$, and $c_{2f} = 0.5$ are determined based on experiment experience. Moreover, the PSO algorithm with the constriction factor (PSO-CK) has been introduced in [27] to guarantee the convergence rate and the search ability, where $w = 0.729$ and $c_1 = c_2 = 1.49$.

Design New Learning Strategies

In addition to the adaptation of control parameters, some researchers have focused on designing different learning strategies with hope to alleviate premature convergence, see e.g. [91, 160, 192, 149, 44, 140, 188, 93, 199, 144]. With the newly designed topological structures, the variant PSO algorithms may possess better population diversity or convergence than the standard PSO algorithm [109, 5, 90, 127]. For example, a comprehensive learning particle swarm optimizer [93] has been developed to preserve the population diversity of the particles to avoid trapping in the local optima for complex multimodal problems. In [192], an adaptive PSO (APSO) algorithm has been proposed with the introduction of an evolutionary factor to distinguish among four evolutionary states and, with this learning strategy, the control parameters have been adaptively adjusted for the PSO algorithm. In [149], a switching PSO (SPSO) algorithm has been put forward to improve the convergence rate by updating the acceleration coefficients based on the switching of different evolutionary states.

Recently, a competitive swarm optimizer has been designed in [25] for large-scaled optimization problems where a pairwise competition mechanism is designed. With this pairwise competition mechanism, the particle that loses the competition adjusts the position according to the winner particle. More recently, time-delay terms have been taken into account through the velocity updating process due to the utilization of historical information during the evolution process, which results in a better accuracy than the standard PSO algorithm, see e.g. [140, 188]. Moreover, the time-delay terms consist of the historical information of the population evolution and the time-delayed PSO algorithms are then likely to have better accuracy than the classic PSO algorithm. It should also be mentioned that an augmented PSO algorithm in combination with multiple adaptive methods has been put forward in [76] with purpose of improving the diversity and avoiding the premature convergence problem, where an intelligent selection mechanism has been put forward to select an appropriate search approach. Additionally, the PSO algorithm with a dynamical diversity coefficient has been proposed in [59] where a random velocity controlled by a diversity coefficient has been taken into consideration to further improve the PSO algorithm by enhancing the search ability.

Hybridize Particle Swarm Optimization with Other Evolutionary Computation Algorithms

In the past few years, the traditional PSO algorithms have been improved in combination with the usage of some popular EC approaches such as the differential evolution (DE) algorithm [186, 187, 201] and the GA [61, 178]. Motivated by the success of other EC algorithms, it seems natural to hybridize other heuristic algorithms with the PSO algorithm. For instance, a switching local evolutionary PSO algorithm has been proposed in [187] by employing the DE algorithm to improve the search ability of the particles and increase the possibility of escaping from the local optima. A hybrid PSO-GA algorithm has been proposed in [54, 57] where the genetic operators (e.g. crossover and mutation) are exploited to balance the global and local searching through the entire search space, and therefore ensure the satisfactory search ability of the particles.

2.3 Applications of Particle Swarm Optimization in Intelligent Data Analysis

In this section, the applications of PSO algorithms in IDA are reviewed (in terms of data clustering and deep learning). Data analysis, as an important research forefront in both academy and industry, is the process of inspecting, transforming, cleaning, and modelling data with the purpose of discovering useful information, informing conclusions, and supporting decision-making. Various statistical and machine learning techniques have been studied for data analysis during the past few decades. In recent years, the introduction of IDA has attracted an ever-increasing research interest in the computer science community. The objective of IDA is to reveal and indicate significant features of a massive amount of data [14, 24, 70]. In this thesis, we aim to review the applications of PSO algorithms in data clustering and deep learning.

2.3.1 Data Clustering

Recognized as a research front with data analysis, clustering techniques have been successfully employed in a variety of research areas such as biology, signal processing, computer vision, market segmentation, and healthcare, see e.g., [146, 141, 52, 152, 39]. Clustering techniques are used to discover the natural groupings of a set of objects where the objects in the same cluster share similar characteristics.

Different starting points and criteria will lead to different taxonomies of clustering algorithms. It has been shown in [32, 119] that many popular clustering algorithms are heavily dependent on the initial state of cluster centroids, and may get trapped in local optima. As such, it is reasonable to *optimize* the parameters of clustering algorithms (e.g. the number of clusters and the initial state of cluster centroids) in order to improve the clustering performance. In this context, various optimization algorithms have been applied to optimally set the parameters with examples including the GA [84, 63], the SA algorithm [133, 128], the PSO algorithm [153, 78], and the artificial bee colony [195] algorithm.

PSO algorithms have proven to be a strong competitor to other optimization algorithms [6, 153, 173, 77]. For instance, a PSO-based clustering technique has been

proposed in [153] where the initial swarm adopts clusters formed by the K-means clustering algorithm. A novel PSO-based clustering algorithm has been developed in [173] for gene clustering by employing the self-organizing map algorithm. Later on, a hybrid fuzzy clustering algorithm based on the conventional PSO algorithm and fuzzy C-means clustering algorithm has been proposed in [77] with satisfactory performance on several well-known benchmark data sets. Very recently, a density-based PSO algorithm has been introduced in [6] for data clustering by combining the kernel density estimation method with the PSO algorithm.

2.3.2 Deep Learning

Machine learning techniques have been widely applied in a variety of areas such as pattern recognition, natural language processing (NLP) and computational learning. With machine learning techniques, computers are endowed with the capability of acting without being explicitly programmed, constructing algorithms that can learn from data, and making data-driven decisions or predictions [96, 132, 179, 62, 12, 118]. Nevertheless, when it comes to the human information processing mechanisms (e.g. speech and vision), the performance of traditional machine learning techniques is far from satisfactory. Inspired by deep hierarchical structures of human speech perception and production systems, the concept of deep learning algorithms has been introduced in the late 20th century. The past few decades have witnessed rapid developments of deep learning techniques with significant impacts on various research areas, such as speech recognition, NLP, information retrieval, compute vision, and image analysis [87, 120, 170, 102, 1]. In fact, due to their strong ability to handle large amounts of unlabeled data, deep learning techniques have drawn tremendous interest from both academic and industrial areas. It is also worth mentioning that the deep learning techniques have attracted the attention of many high-tech enterprises such as Google, Facebook and Microsoft. For example, in March 2016, a Go Game match was held in South Korea by Google's deep learning team (called DeepMind) between its AI player AlphaGo and one of the world's strongest players Lee Se-dol [138].

The concept of deep learning originates from the study on artificial neural networks (ANNs) [74]. ANNs have become an active research area during the past few decades

[198, 175, 180, 182, 26]. To construct a standard neural network (NN), it is essential to utilize neurons to produce real-valued activations and, by adjusting the weights, the NNs behave as expected. However, depending on the problems, the process of training a NN may take long causal chains of computational stages. Backpropagation is an efficient gradient descent algorithm which has played an important role in NNs since 1980 [132]. It trains the ANNs with a teacher-based supervised learning approach. Although the training accuracy is high, the performance of the backpropagation algorithm when applied to the testing data might not be satisfactory. As the backpropagation algorithm is based on local gradient information with a random initial point, the algorithm is often trapped in local optima. Furthermore, if the size of the training data is not big enough, NNs will face the problem of overfitting. Consequently, other effective machine learning algorithms such as support vector machine (SVM), boosting and K-nearest neighbor (KNN) algorithms have been adopted to obtain global optimum with lower power consumption. In 2006, Hinton [73] proposed a new training method (called layer-wise-greedy-learning) which marked the birth of deep learning techniques. The basic idea of the layer-wise-greedy-learning is that unsupervised learning should be performed for network pre-training before the subsequent layer-by-layer training. By extracting features from the inputs, the data dimension is reduced and a compact representation is hence obtained. Then, exporting the features to the next layer, all of the samples will be labeled and the network will be fine-tuned with the labeled data. The reason for the popularity of deep learning is twofold: on one hand, the development of big data analysis techniques indicates that the overfitting problem in training data can be partially solved; on the other hand, the pre-training procedure before unsupervised learning will assign non-random initial values to the network. Therefore, a better local minimum can be reached after the training process and a faster converge rate can be achieved.

With rapid development of computation techniques, a powerful framework has been provided by ANNs with deep architectures for supervised learning. Generally speaking, the deep learning algorithm consists of a hierarchical architecture with many layers, each of which constitutes a non-linear information processing unit. In this thesis, we only discuss deep architectures in NNs. Deep neural networks (DNNs), which

employ deep architectures in NNs, can represent functions with higher complexity if the numbers of layers and units in a single layer are increased. Given enough labeled training datasets and suitable models, deep learning approaches can help humans establish mapping functions for operation convenience.

The DNNs has been widely applied in speech recognition and acoustic modeling for audio classification [88]. Besides, deep learning approaches also play an important role in the area of image processing such as handwritten classification [83], high-resolution remote sensing scene classification [75], single image super-resolution [42], and multi-category rapid serial visual presentation brain computer interfaces [110]. Moreover, deep architectures have also been employed in multi-task learning for NLP with an enhanced inference robustness [29, 86].

Although deep learning techniques have been successfully applied to a variety of applications, the problem of choosing suitable hyperparameters has become increasingly significant in deep learning. Note that there are more than hundreds of hyperparameters in some industrial applications using DNNs such as NLP and video detection, adding enormous challenges to select appropriate hyperparameters. Generally speaking, it requires a large amount of experience to choose an appropriate value of numerical hyperparameters, e.g. the learning rate, the number of hidden units, the momentum, and the weight decay [123, 126, 13, 53, 124, 196]. Under this circumstance, it is of vital importance to design effective approaches for hyperparameter selection in DNNs. Owing to their outstanding performance in discovering optimal solutions, evolutionary computation approaches become a seemingly natural choice in optimizing the hyperparameters of the DBNs. Recently, a variety of meta-heuristic algorithms (e.g., harmony search, GA and PSO algorithms) have been successfully employed to optimize the hyperparameter selection process in the DNNs. For example, the harmony search algorithm has been used in [123] to optimize the hyperparameters (including the learning rate, the weight decay, the penalty parameter, and the number of hidden units) during the pre-training process. The genetic algorithm has been adopted in [53] to tune the hyperparameters of a deep convolutional neural network. In order to prevent overfitting, the meta-heuristic-driven techniques (including the PSO algorithm,

bat algorithm, cuckoo search, and firefly algorithm) have been utilized to optimize the dropout (a regularization term) in the convolutional neural networks [36].

As mentioned previously, PSO algorithms serve as a powerful family of algorithms that can be effectively used to solve global optimization problems. Intuitively, a natural idea is to introduce PSO algorithms in DNNs with the purpose of selecting proper hyperparameters in an intelligent way [123, 126, 13, 53]. For example, the PSO algorithm has been employed to optimize the hyperparameters in a deep convolutional neural network in [2]. In [126], the PSO algorithm has been utilized to optimize the hyperparameters in a deep Boltzmann machine. A non-linear marginalized stacked denoising autoencoder has been proposed in [143] to extract useful features in visual speech recognition where the PSO algorithm is employed to optimize the hyperparameters of the proposed model. To summarize, the utilization of PSO algorithms in the deep learning community achieves a great success and it seems a natural idea to investigate advanced PSO algorithms for optimizing the hyperparameters of the deep learning architectures.

Chapter 3

A Novel Sigmoid-Function-Based Adaptive Weighted Particle Swarm Optimizer

3.1 Motivation

Optimization problem has long been an important research topic attracting enormous interest from a variety of communities owing to its clear application potential in real-world systems such as telecommunication systems, power systems, and network operating systems [137, 148, 140, 188, 121, 164, 93, 40, 177, 94]. In the past few decades, evolutionary computation (EC) techniques have been successfully employed to effectively solve the optimization problems. In this regard, a famous EC approach, known as the PSO algorithm, has been successfully implemented in various practical applications in dealing with the optimization problems [46]. In a PSO algorithm, as motivated by the swarm intelligence and social behaviors (e.g., birds flocking), all the particles are randomly initialized and then encouraged to explore the problem space thoroughly based on the individual experience and the interaction with other particles [135, 136]. During the evolution process, the historically personal best position (*pbest*) of each particle as well as the historically global best position (*gbest*) discovered by the entire swarm are two important positions, based on which the particles are motivated

to seek the optimal solution. According to [46, 48], the PSO algorithm outperforms many popular EC approaches due to its easy implementation and fast convergence rate.

It is well known that, the balance between global and local searches throughout the searching process plays a vital role in successfully finding the optimal solution [145, 181]. The inertia weight as well as acceleration coefficients, which serve as two control parameters, are vitally important in the velocity updating model of the PSO algorithm and have been extensively investigated in recent years for better accuracy and faster convergence [30, 129, 145, 194, 17]. Up to now, some PSO variants have been focused on the modification of the aforementioned control parameters. In [135, 136], a linear-decreasing-inertia-weight-based PSO (PSO-LDIW) algorithm has been proposed where the inertia weight is updated in a time-varying manner. For the purpose of efficiently controlling the local and global searches, the time-varying-acceleration-coefficient-based PSO (PSO-TVAC) algorithm has been introduced in [129]. In addition to the adaptation of the control parameters, topological structures have been introduced in some PSO algorithms with the hope to alleviate premature convergence, see e.g. [192, 158, 106, 140, 188, 93]. In particular, time-delay terms have been taken into account through the velocity updating process due to their utilization of historical information during the evolution process which results in a better accuracy than the standard PSO algorithm, see e.g. [140, 188].

Although some popular PSO algorithms have exhibited competitive performance on searching the global optimum and increasing the possibility of avoiding the local optima, the enhancement of the search performance of PSO algorithms is often at the expense of sacrificing the convergence rate, which is certainly undesirable [25, 30, 23]. As such, it is of practical significance to develop a new PSO algorithm that is capable of finding the globally optimal solution yet with a *satisfactory* convergence rate through adaptively updating the control parameters. Note that the inertia weight and acceleration coefficients only change along with time in most of the existing PSO algorithms. In this case, a seemingly natural idea is to make full use of the distances from each individual particle to its *pbest* and *gbest* at *each iteration*, and adaptively update the control parameters according to the outputs of a certain sigmoid function with the calculated

distances as the inputs. In comparison with the time-varying parameter strategy (see e.g. [135, 136, 129]), the sigmoid-function-based updating strategy possesses the following advantages: 1) the control parameters are adaptively chosen which could guarantee the search ability of the optimizer; and 2) the particles are motivated to move towards the *pbest* and *gbest* as fast as necessary which could help improve the convergence rate. It should be mentioned that the particles slow down once they get close to the *pbest* and *gbest*.

To summarize, the main objective of this chapter is to propose an adaptive weighting (AWPSO) algorithm with a sigmoid-function-based parameter selection scheme. *The main contributions are outlined as follows: 1) a novel sigmoid-function-based AWPSO algorithm is proposed where an adaptive weighting strategy is designed to adaptively adjust the control parameters at each iteration; and 2) The acceleration coefficients are adaptively controlled according to the distances from the particle to its pbest and gbest, thereby facilitating a relatively fast exploitation of the problem space.*

The rest of this chapter is organized below. Section 3.2 describes the proposed adaptive weighting strategy and the AWPSO algorithm. Benchmark functions, test PSO algorithms, parameter setting, experiment results and discussions are illustrated in Section 3.3. Conclusions and future directions are presented in Section 3.4.

3.2 A Novel AWPSO Algorithm

In a PSO algorithm, the acceleration coefficients are used to motivate the particles to move to the *pbest* and *gbest*. The distances from the position of each particle to its *pbest* and *gbest* are dominantly important in determining the movement of the particles. On the other hand, the adaptation of the control parameters is a significant factor in seeking the optimal solution with convincing efficiency and accuracy [129, 35]. Therefore, to control the PSO algorithm in an effective way, in this chapter, we endeavor to propose a novel adaptive weighting mechanism with which the acceleration coefficients are adaptively adjusted as the iteration goes.

3.2.1 Adaptive Weighting Strategy

In the classic PSO algorithm, the velocity of an individual particle gets accelerated according to the *distances* from the particle to its *pbest* and *gbest*. As such, the selection of appropriate acceleration coefficients is of vital importance for finding the globally optimal solution through the problem space. In this case, it makes both theoretical and practical sense to *adaptively* updating the acceleration coefficients iteration by iteration based on the aforementioned *distances* to efficiently improve the searching capability of the PSO algorithm.

In the literature, several popular updating strategies for acceleration coefficients have been proposed during the past decade with satisfactory performance [129] while avoiding premature convergence. Another PSO variant with linearly decreasing strategy has been developed in [142] to update acceleration coefficients. However, these PSO variants only adjust the acceleration coefficients in a time-varying manner without taking the information of the population evolution into account.

It is clear that all the individuals are encouraged to explore the entire search space as much as possible in the early stage of the evolution process. Then, in the latter stage of the optimization process, the individuals are motivated to converge to the global optimum and find the optimization solution as fast as possible. As can be seen in Eq. (2.1), the velocity of the particle updates is dependent on the distances from the particles to their own *pbest* and the *gbest*. In this case, it is reasonable to adjust the acceleration coefficient according to the distances from each individual particle to its *pbest* as well as the *gbest*.

Taking above all the mentioned concerns into consideration, an adaptive weighting strategy is proposed to adaptively control the acceleration coefficients. The main motivation is to accelerate the particles to find the optimal solution as fast as possible and thus enhances the convergence rate. Different from the time-varying updating strategy, the acceleration coefficients are altered according to the distance of the particle towards its *gbest* and *pbest*. If the particle is far away from its *pbest* and *gbest*, a relatively large acceleration coefficient is employed to accelerate the particle. However, the value of the acceleration coefficient is limited in an appropriate range to avoid

premature convergence, which means that the velocity should be bounded to guarantee the searching capability of the algorithm.

Motivated by above discussions, we believe that an adaptive weighting updating function is appropriate to describe the relationship between the acceleration coefficient and the distances (from the particle to its *pbest* and *gbest*). In other words, the updates of the former acceleration coefficients should be *adaptive* to the latter distances, thereby fully justifying the velocity of the particle movements towards the global optimum. From a mathematical viewpoint, the proposed adaptive weighting updating rule can be described as follows:

$$\begin{aligned} c_{g_{pi}}(k) &= F(g_{pi}(k)) \\ c_{g_{gi}}(k) &= F(g_{gi}(k)) \end{aligned} \tag{3.1}$$

where the function $F(\cdot)$ represents the adaptive weighting updating function to be discussed later; and $g_{pi}(k)$ and $g_{gi}(k)$ are defined by

$$\begin{aligned} g_{pi}(k) &= p_i(k) - x_i(k) \\ g_{gi}(k) &= p_g(k) - x_i(k), \end{aligned} \tag{3.2}$$

which denote the distances from the particle i to its *pbest* and *gbest* at the kth iteration, respectively.

3.2.2 Selection of Adaptive Weighting Updating Function

Intuitively, the adaptive weighting updating function should have the following two properties: 1) the updating function is monotonically increasing; and 2) the updating function is bounded. The first property is mainly due to the characteristics of the acceleration coefficients. It is well known that the acceleration coefficients are the weighting terms which pull the particles to the *pbest* and *gbest*. A particle which is far away from its *pbest* and *gbest* requires a fast movement towards its *pbest* and *gbest*. Therefore, a monotonically increasing function is required. The second property is justified by the fact that the search space of a constrained optimization problem is normally bounded. Once a particle is close to its *pbest* and *gbest*, the movement should

be slowed down to avoid missing its *pbest* and the *gbest*. Consequently, the acceleration coefficients should be bounded for the control of the velocity of the particle.

In search of adequate updating functions that are both monotonically increasing and uniformly bounded, the activation functions employed in neural networks appear to be ideal candidates. There are some popular activation functions for the neural networks such as step functions and sigmoid functions, among which we decide to select the sigmoid function as the adaptive weighting updating function for three reasons: 1) the sigmoid function is monotonic and bounded; 2) the curve of the sigmoid function is S-shaped and this would avoid undesirable abrupt changes of the control parameters; and 3) the sigmoid function is smooth and differentiable, thereby reflecting the adaptive/dynamic nature of the weight updating iteration by iteration.

According to the above discussion, in this chapter, a sigmoid function is employed to adjust the acceleration coefficients as follows:

$$F(D_p) = \frac{b}{1 + e^{-a \times (D_p - c)}} + d \tag{3.3}$$

where e is the natural logarithm base; a denotes the steepness of the curve which is a constant value; b represents the peak value of the curve; c represents the abscissa value of the central point of the curve; d is a positive constant value; and D_p is the input of the function which is determined by Eq. (3.2). Specifically, D_p is the distance between the particle and its *pbest* for the cognitive acceleration coefficient. For the social acceleration coefficient, D_p indicates the distance between the particle and the *gbest*.

Remark 1: In Eq. (3.3), it is of vital importance to choose appropriate values of the four parameters (a, b, c and d). Note that a is the parameter which denotes the steepness of the curve. It seems a natural idea to adjust the value of a according to the search range of each individual optimization problem. In our work, we have run 25 tests to determine an appropriate value of each parameter. According to empirical studies, we set $a = 0.000035 \cdot m$ where m indicates the search range of the optimization problem. According to the characteristics of the sigmoid function and experimental experience, b, c, d are set to be 0.5, 0, and 1.5, respectively.

To conclude, the three major advantages of the proposed sigmoid-function-based adaptive weighting strategy are summarized as follows:

1. the acceleration coefficients are adaptively controlled within reasonable bounds, and the adaptive weighting strategy ensures the efficiency of the velocity updating process;

2. the adaptive weighting updating function, chosen as the sigmoid function, is utilized to reflect the monotonic yet relatively smooth changes of the acceleration coefficients, where a larger distance will lead to a larger value of acceleration coefficient; and

3. the particles are motivated to seek the optimal solution as fast as *necessary*, thereby improving the convergence rate.

3.2.3 Framework of the AWPSO Algorithm

An AWPSO algorithm is developed in this chapter where the velocity updating equation obeys an adaptive weighting strategy. During the population evolution process, the velocity and position of the ith particle are updated on the basis of the following equations:

$$\begin{aligned} v_i(k+1) &= w \times v_i(k) + c_{g_{pi}}(k) \times r_1 \times g_{pi}(k) + c_{g_{gi}}(k) \times r_2 \times g_{gi}(k) \\ x_i(k+1) &= x_i(k) + v_i(k+1) \end{aligned} \tag{3.4}$$

where w is the inertia weight; $g_{pi}(k)$ and $g_{gi}(k)$ represent the distances from the particle i to its *pbest* and *gbest* at the kth iteration, respectively; $c_{g_{pi}}(k)$ denotes the acceleration constant determined by $g_{pi}(k)$, and $c_{g_{gi}}(k)$ indicates the acceleration constant determined by $g_{gi}(k)$.

The flowchart of the introduced AWPSO algorithm is depicted in Fig. 3.1.

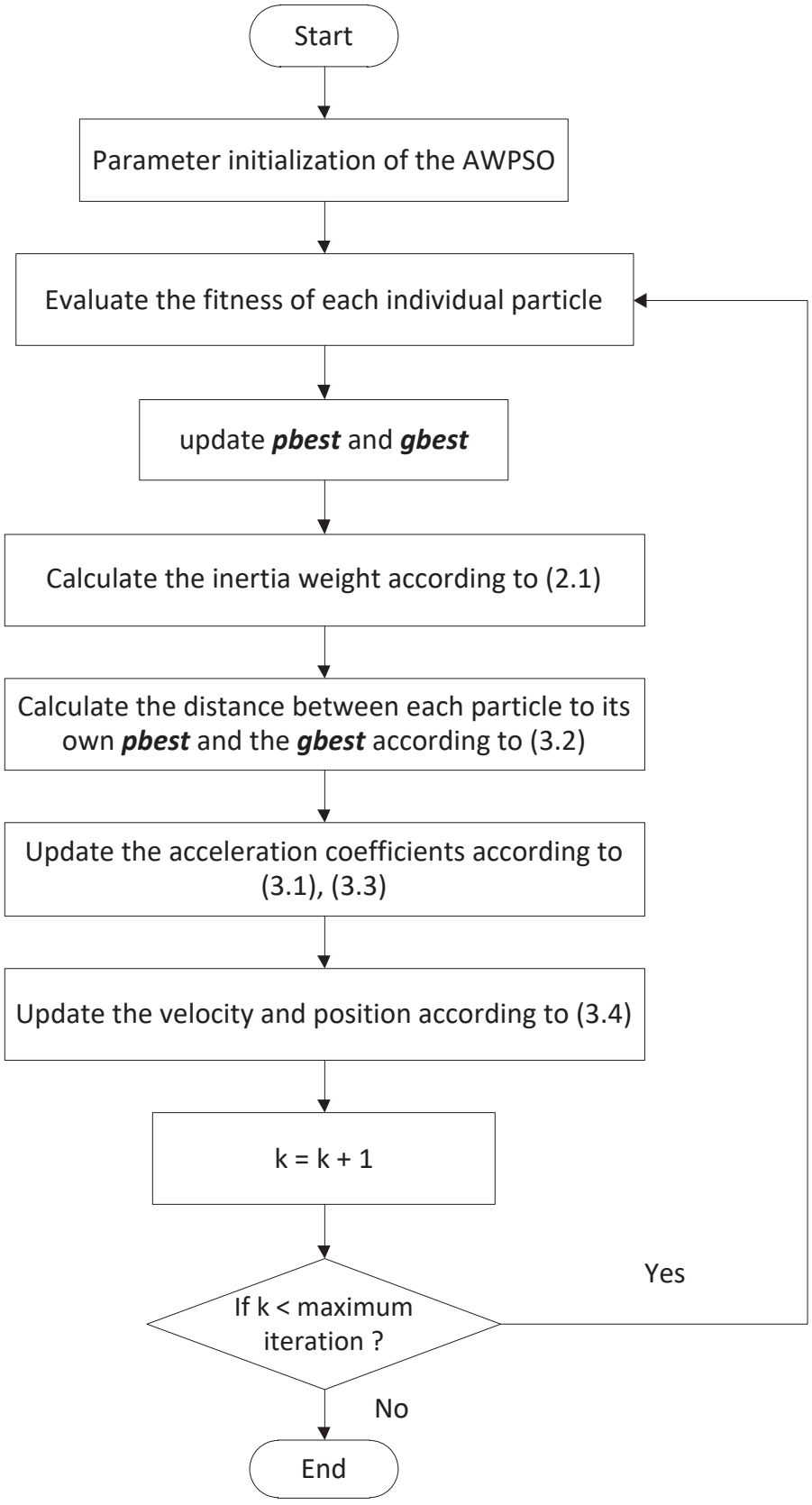

Fig. 3.1 Flowchart of the AWPSO algorithm

3.3 Results and Discussions

In our work, the AWPSO algorithm is compared with some popular variant PSO algorithms on a series of widely-used optimization benchmark functions consisting of both unimodal and multimodal cases for performance evaluation. In addition, the convergence performance of the adaptive weighting updating function is demonstrated with visible results. For all the benchmark functions, the swarm size is set to be 30 and the dimension of the problem space is set to be 30. In this simulation, each experiment has been repeated for 50 times independently, and the maximum iteration number is set to be 5000. It is worth pointing out that the Euclidean distance is chosen as the distance metric in the simulation.

The performance indicators are of vital importance in evaluating the performance of the proposed PSO algorithms. In general, the convergence rate, the solution accuracy, the successful convergence ratio (also known as the success ratio), and the population diversity are four major performance indicators to evaluate evolutionary computation approaches. In this thesis, the convergence rate is employed to evaluate the convergence performance of the PSO algorithm. The solution accuracy is used to verify the effectiveness of the discovered solution, which is measured by the mean, the minimum, and the standard deviation of the fitness value for all the benchmark functions. The success ratio is used to describe the convergence accuracy of the PSO algorithms. In addition, the population diversity of the propose PSO algorithm is justified by using the formula defined in Eq. (3.13).

3.3.1 Benchmark Functions

It should be noticed that all the selected benchmark functions have been widely used in the evolutionary computing community [176, 188, 140]. The Sphere function $f_1(x)$ is a typical unimodal function. The Rosenbrock function $f_2(x)$ is called as the Rosenbrock's banana function which is a popular benchmark function. The Rastrigin function $f_3(x)$, the Penalized 1 function $f_5(x)$ and the Penalized 2 function $f_8(x)$ are classical multimodal problems consisting of many local optima, which are difficult to find the globally optimal solution. The Schwefel 2.22 function $f_4(x)$ and the Step function

$f_6(x)$ are also frequently used benchmark functions for optimization. The Schwefel function $f_7(x)$ is a typical benchmark function with lots of local minima. Detailed information of the chosen benchmark functions is given by (3.5) to (3.12). Note that $x = (x_1, x_2, \cdots, x_D)$ where D represents the dimension of the search space, and we set $D = 30$ in the simulation.

$$\text{Sphere}: f_1(x) = x_1^2 + x_2^2 + \cdots + x_D^2. \tag{3.5}$$

$$\text{Rosenbrock}: f_2(x) = \sum_{i=1}^{D-1} \Big[100(x_{i+1} - x_i)^2 + (x_i - 1)^2\Big]. \tag{3.6}$$

$$\text{Rastrigin}: f_3(x) = \sum_{i=1}^{D}(x_i^2 - 10\cos 2\pi x_i + 10). \tag{3.7}$$

$$\text{Schwefel 2.22}: f_4(x) = \sum_{i=1}^{D} \mid x_i \mid + \prod_{i=1}^{D} \mid x_i \mid . \tag{3.8}$$

$$\begin{aligned}\text{Penalized 1}: f_5(x) = \frac{\pi}{D}\Big(&10\sin^2(\pi y_1) \\ &+ \sum_{i=1}^{D-1}(y_i - 1)^2\Big(1 + 10\sin^2(\pi y_{i+1})\Big) \\ &+ (y_D - 1)^2\Big) + \sum_{i=1}^{D} u(x_i).\end{aligned} \tag{3.9}$$

$$y_i = 1 + 1/4(x_i + 1),$$

$$u(x_i) = \begin{cases} 100(-x_i - 10)^4, & x_i < -10, \\ 0, & |x_i| \leq 10, \\ 100(x_i - 10)^4, & x_i > 10. \end{cases}$$

$$\text{Step}: f_6(x) = \sum_{i=1}^{D}(\lfloor x_i + 0.5 \rfloor)^2. \tag{3.10}$$

$$\begin{aligned}\text{Schwefel}: f_7(x) = &418.9828872724338D \\ &- \sum_{i=1}^{D} x_i \sin(\sqrt{|x_i|}).\end{aligned} \tag{3.11}$$

$$\text{Penalized 2}: f_8(x) = 0.1\Big(\sin^2(3\pi x_1) \tag{3.12}$$

$$
\begin{aligned}
&+ \sum_{i=1}^{D-1} (x_i - 1)^2 \Big(1 + \sin^2(3\pi x_{i+1})\Big) \\
&+ (x_D - 1)^2 \Big(1 + \sin^2(2\pi x_D)\Big)\Big) \\
&+ \sum_{i=1}^{D} u(x_i).
\end{aligned}
$$

$$
u(x_i) = \begin{cases} 100(-x_i - 5)^4, & x_i < -5, \\ 0, & |x_i| \leq 5, \\ 100(x_i - 5)^4, & x_i > 5. \end{cases}
$$

The configurations of the benchmark functions are presented in Table 3.1. The search range represents the range of the search space. Additionally, the threshold is a problem-based parameter, which is utilized as a stopping criterion of the algorithm.

Table 3.1 Configuration of benchmark functions

Function Number	Function Name	Problem Dimension	Search Range	Minimum	Threshold
$f_1(x)$	Sphere	30	$[-100, 100]$	0	0.1
$f_2(x)$	Rosenbrock	30	$[-30, 30]$	0	100
$f_3(x)$	Rastrigin	30	$[-5.12, 5.12]$	0	50
$f_4(x)$	Schwefel 2.22	30	$[-10, 10]$	0	0.1
$f_5(x)$	Penalized 1	30	$[-50, 50]$	0	0.1
$f_6(x)$	Step	30	$[-100, 100]$	0	0.1
$f_7(x)$	Schwefel	30	$[-500, 500]$	0	0.1
$f_8(x)$	Penalized 2	30	$[-50, 50]$	0	0.1

3.3.2 Experiment Results

In this chapter, four currently popular PSO algorithms (including the basic PSO algorithm [46], the PSO-LDIW algorithm [135], the PSO-CK algorithm [27], and the SDPSO algorithm [188]) are selected for performance evaluation via eight widely used benchmark functions.

Experiment results are displayed in Figs. 3.2-3.9 where the vertical coordinate indicates the mean fitness value in the logarithmic form, and the horizontal coordinate indicates the iteration number. From the figures, we can see that the AWPSO algorithm exhibits competitive performance on most of the benchmark functions. Although the PSO-LDIW algorithm obtains better mean fitness value than the AWPSO algorithm

Table 3.2 Algorithm evaluation on eight benchmark functions

		PSO	PSO-LDIW	PSO-CK	SDPSO	AWPSO
$f_1(x)$	Minimum	1.75×10^{3}	2.03×10^{-33}	8.13×10^{-87}	4.11×10^{-3}	5.25×10^{-45}
	Mean	2.72×10^{3}	2.00×10^{2}	6.00×10^{2}	1.2816	4.00×10^{2}
	Std. Dev.	1.44×10^{3}	1.41×10^{3}	2.40×10^{3}	1.9592	1.98×10^{3}
	Ratio	0%	98%	94%	22%	96%
$f_2(x)$	Minimum	2.70×10^{5}	2.49×10^{-2}	1.51×10^{-4}	7.77×10^{1}	3.95×10^{-2}
	Mean	4.66×10^{5}	1.28×10^{4}	5.54×10^{3}	2.29×10^{3}	5.90×10^{3}
	Std. Dev.	1.07×10^{5}	3.15×10^{4}	2.16×10^{4}	1.27×10^{4}	2.15×10^{4}
	Ratio	0%	74%	88%	4%	68%
$f_3(x)$	Minimum	1.68×10^{2}	1.19×10^{1}	4.88×10^{1}	2.97×10^{1}	2.39×10^{1}
	Mean	2.00×10^{2}	4.64×10^{1}	9.46×10^{1}	6.47×10^{1}	5.70×10^{1}
	Std. Dev.	1.75×10^{1}	2.31×10^{1}	2.59×10^{1}	2.42×10^{1}	2.27×10^{1}
	Ratio	0%	62%	2%	36%	42%
$f_4(x)$	Minimum	1.89×10^{1}	6.32×10^{-22}	1.73×10^{-26}	1.13×10^{-2}	1.98×10^{-16}
	Mean	3.96×10^{1}	2.62×10^{1}	1.06×10^{1}	8.6727	2.32×10^{1}
	Std. Dev.	1.47×10^{1}	1.82×10^{1}	1.02×10^{1}	1.41×10^{1}	1.63×10^{1}
	Ratio	0%	10%	36%	4%	12%
$f_5(x)$	Minimum	1.81×10^{1}	1.57×10^{-32}	1.57×10^{-32}	1.08×10^{-4}	1.57×10^{-32}
	Mean	9.23×10^{1}	8.29×10^{-3}	2.77×10^{-1}	2.38×10^{-1}	2.70×10^{-2}
	Std. Dev.	1.47×10^{2}	2.84×10^{-2}	4.44×10^{-1}	3.37×10^{-1}	4.59×10^{-2}
	Ratio	0%	92%	46%	54%	74%
$f_6(x)$	Minimum	1.62×10^{3}	0.0000	0.0000	0.0000	0.0000
	Mean	3.17×10^{3}	2.00×10^{-2}	1.01×10^{3}	4.7400	4.00×10^{-2}
	Std. Dev.	2.40×10^{3}	1.41×10^{-1}	3.03×10^{3}	4.4758	1.98×10^{-1}
	Ratio	0%	98%	14%	14%	96%
$f_7(x)$	Minimum	4.78×10^{3}	1.90×10^{3}	3.22×10^{3}	3.21×10^{3}	1.54×10^{3}
	Mean	6.60×10^{3}	3.64×10^{3}	4.90×10^{3}	5.06×10^{3}	3.70×10^{3}
	Std. Dev.	1.03×10^{3}	1.55×10^{3}	8.85×10^{2}	1.30×10^{3}	2.23×10^{3}
	Ratio	0%	0%	0%	0%	0%
$f_8(x)$	Minimum	2.84×10^{4}	4.18×10^{-32}	1.35×10^{-32}	2.21×10^{-2}	1.35×10^{-32}
	Mean	1.35×10^{5}	2.42×10^{-3}	2.02×10^{-1}	5.49×10^{-1}	3.07×10^{-3}
	Std. Dev.	6.90×10^{4}	7.12×10^{-3}	6.19×10^{-1}	4.56×10^{-1}	8.59×10^{-3}
	Ratio	0%	100%	84%	10%	100%

on most of the benchmark functions, the superiority is not obvious. Furthermore, it is apparent that the AWPSO algorithm converges faster than most of the benchmark functions with satisfactory mean fitness value.

In our work, the diversity of the swarm at the kth iteration is calculated as follows [151]:

$$S(k) = \frac{1}{M} \sum_{i=1}^{M} \sqrt{\sum_{j=1}^{D} \left(x_{ij}(k) - \bar{x}_j(k)\right)^2} \tag{3.13}$$

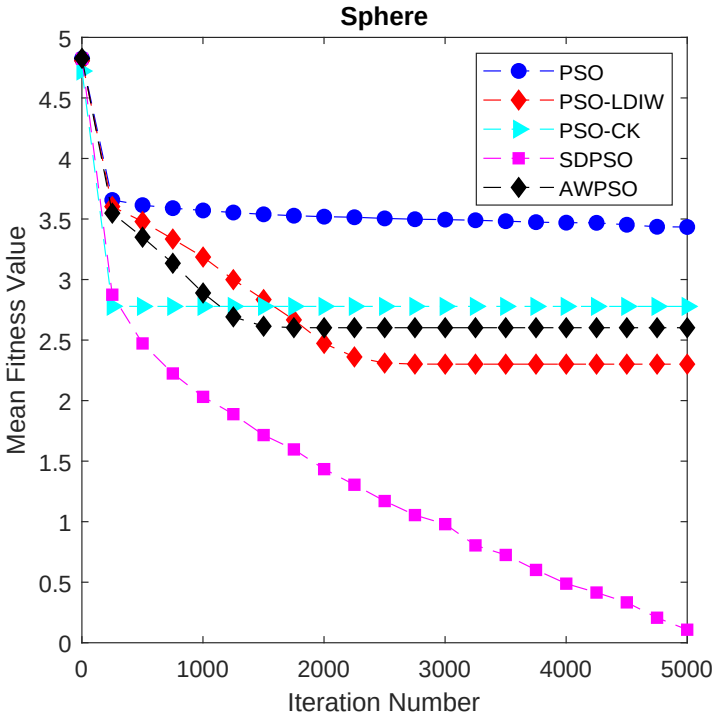

Fig. 3.2 Optimization performance for Sphere function $f_1(x)$

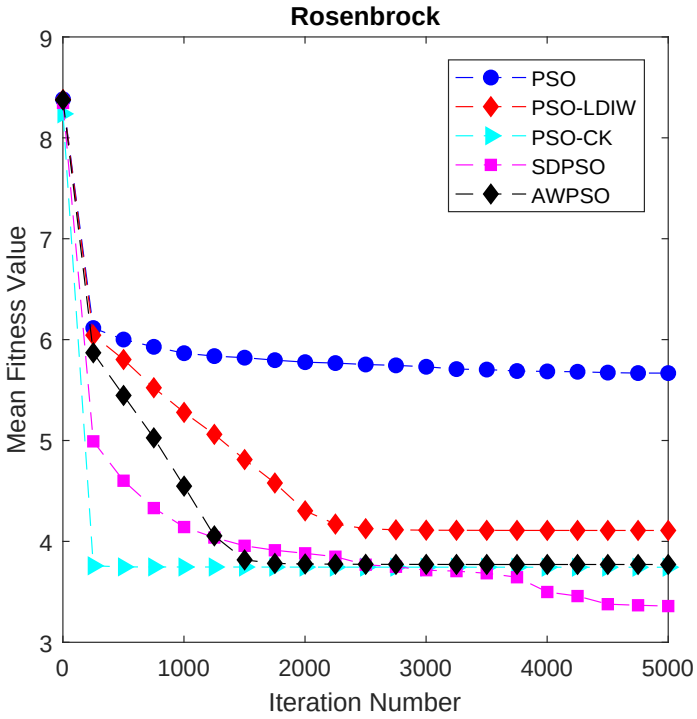

Fig. 3.3 Optimization performance for Rosenbrock function $f_2(x)$

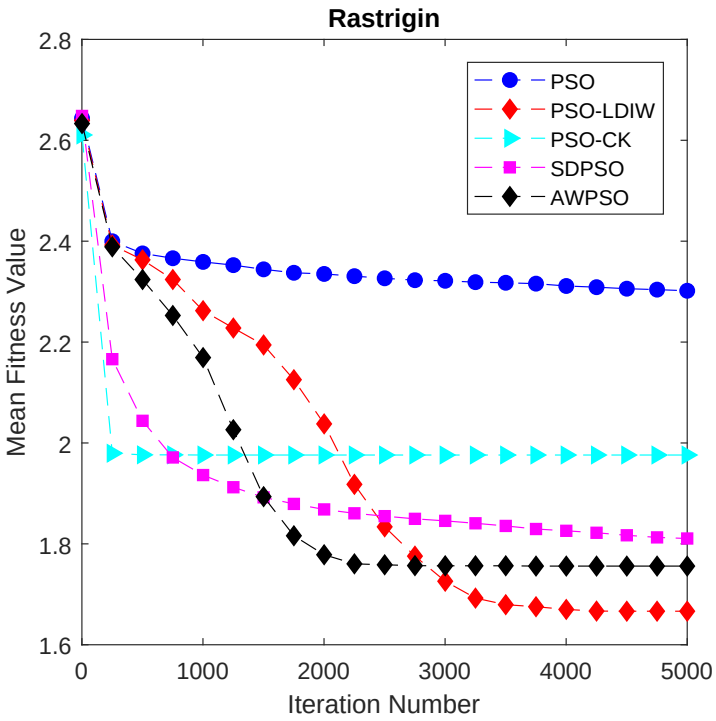

Fig. 3.4 Optimization performance for Rastrigin function $f_3(x)$

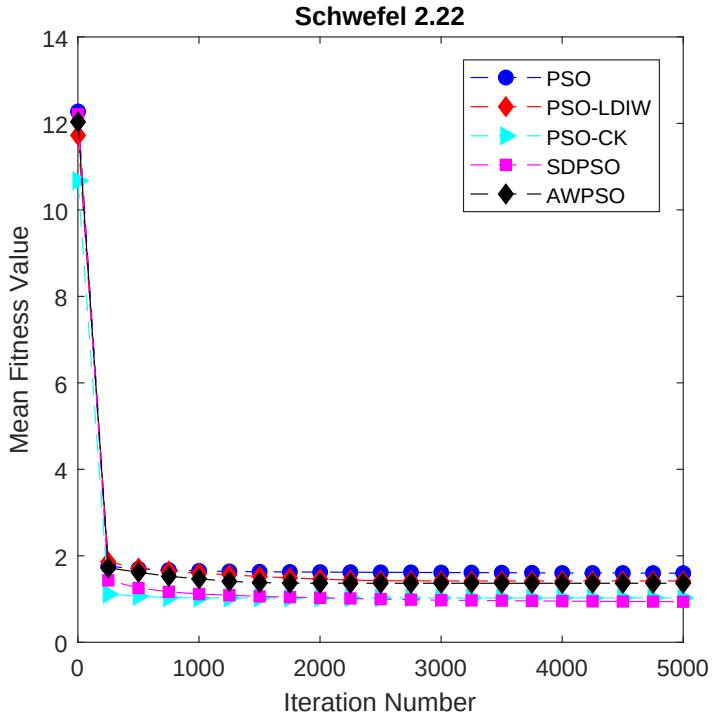

Fig. 3.5 Optimization performance for Schwefel 2.22 function $f_4(x)$

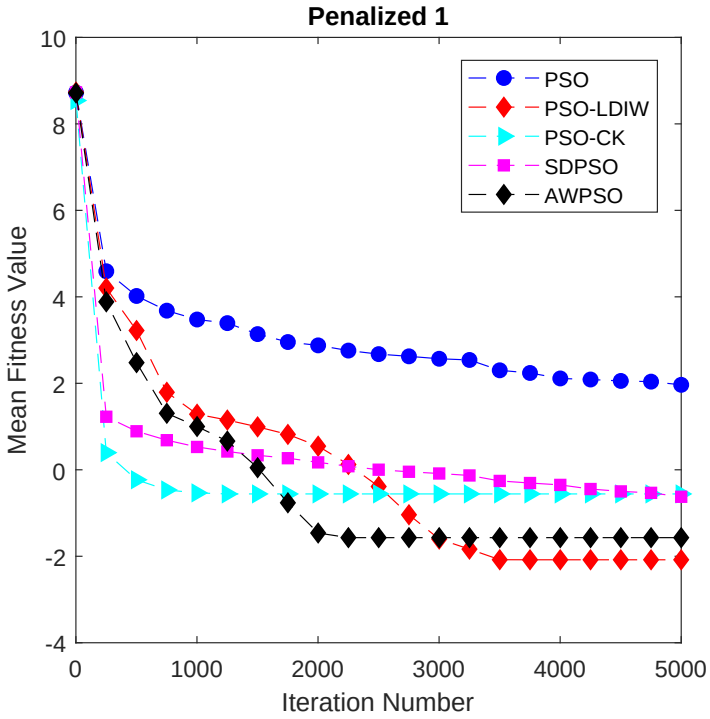

Fig. 3.6 Optimization performance for Penalized 1 function $f_5(x)$

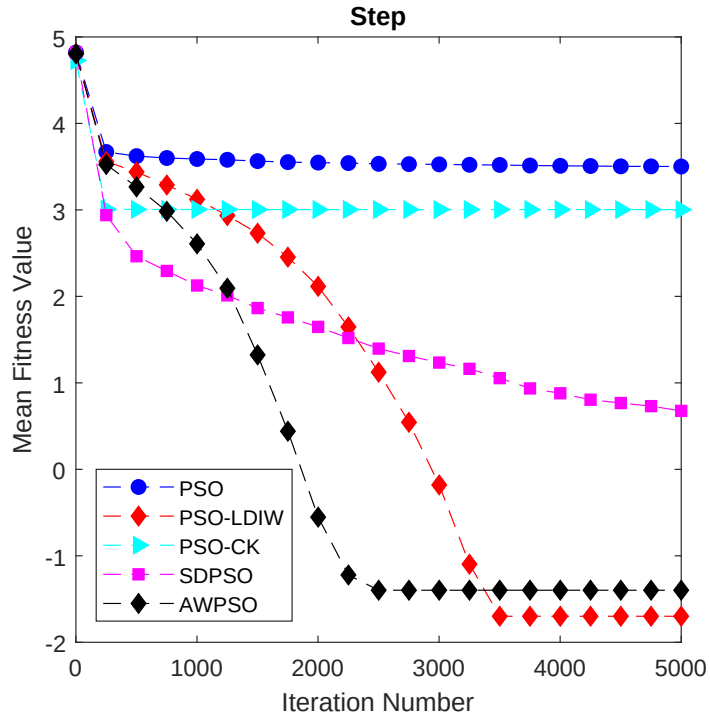

Fig. 3.7 Optimization performance for Step function $f_6(x)$

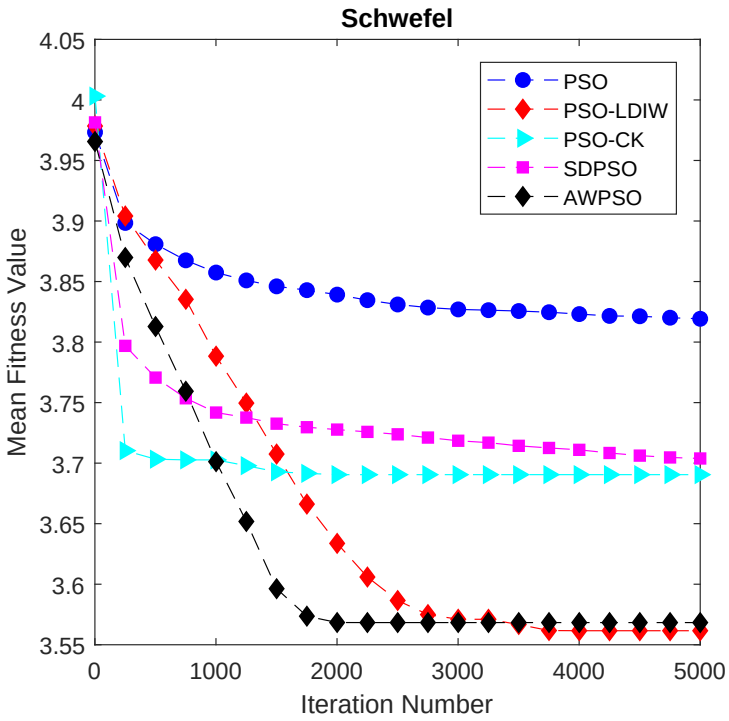

Fig. 3.8 Optimization performance for Schwefel function $f_7(x)$

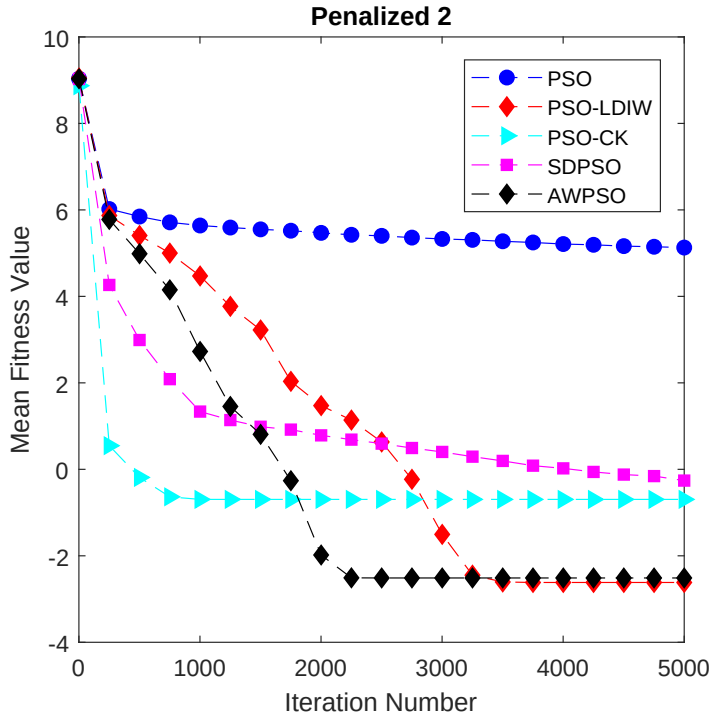

Fig. 3.9 Optimization performance for Penalized 2 function $f_8(x)$

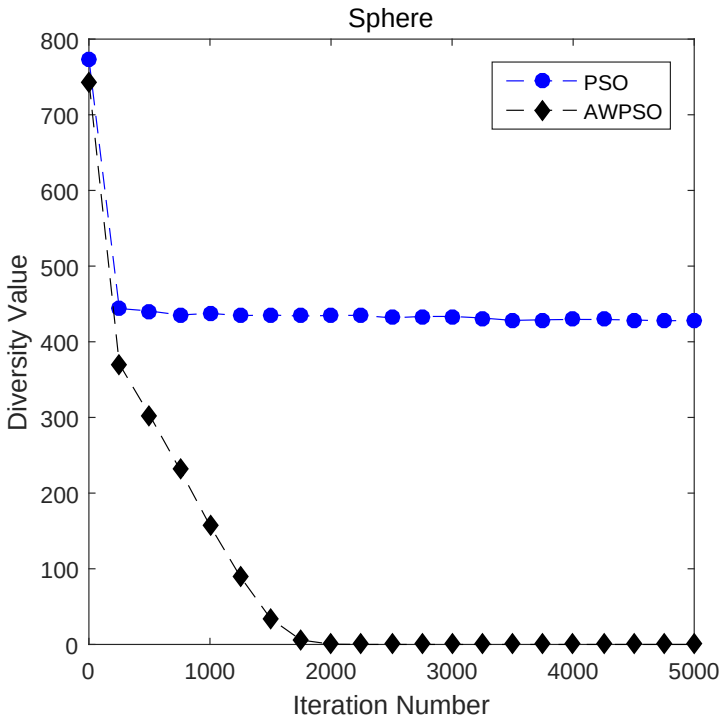

Fig. 3.10 Diversity measure for Sphere function $f_1(x)$

where M is the swarm size, D is the dimensionality of the optimization problem, x_{ij} denotes the ith particle at the jth dimension, $\bar{x}_j(k)$ is the average value of the jth dimension over all particles at the kth iteration, i.e. $\bar{x}_j(k) = \frac{1}{M}\sum_{i=1}^{M} x_{ij}(k)$.

The population diversity of the classic PSO algorithm and our proposed AWPSO algorithm are shown in Figs. 3.10-3.17, where the vertical coordinate represents the diversity measure of the swarm and the horizontal coordinate indicates the number of iteration. It can be seen that both the classic PSO algorithm and the AWPSO algorithm have large values of population diversity at the early stage of the optimization process. The population diversity of the classic PSO algorithm and the AWPSO algorithm decreases as the iteration number increases. It is worth mentioning that a small value of population diversity implies that the population converges to a certain region of the search space. We can see that the population diversity of the AWPSO algorithm is smaller than that of the classic PSO algorithm at the later stage of the optimization process, which indicates that the convergence of the AWPSO algorithm is better than the classic PSO algorithm.

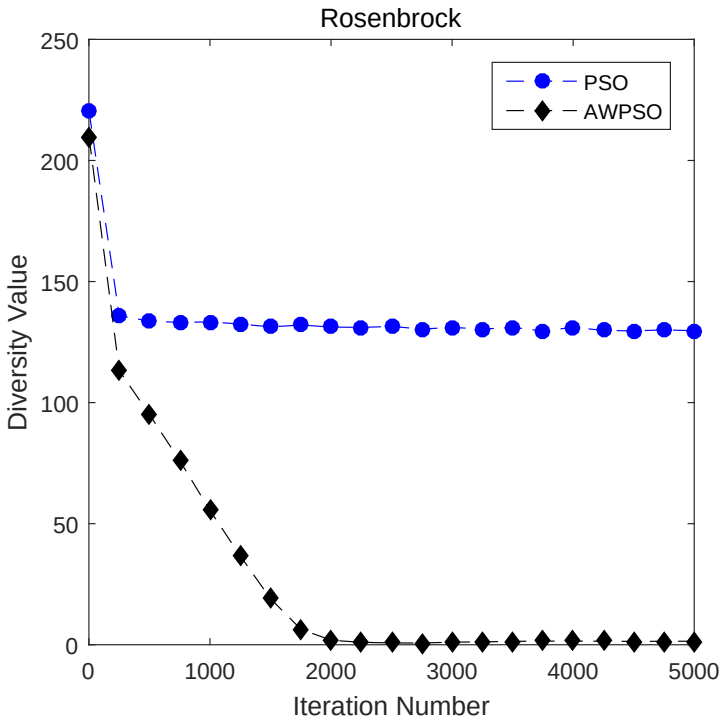

Fig. 3.11 Diversity measure for Rosenbrock function $f_2(x)$

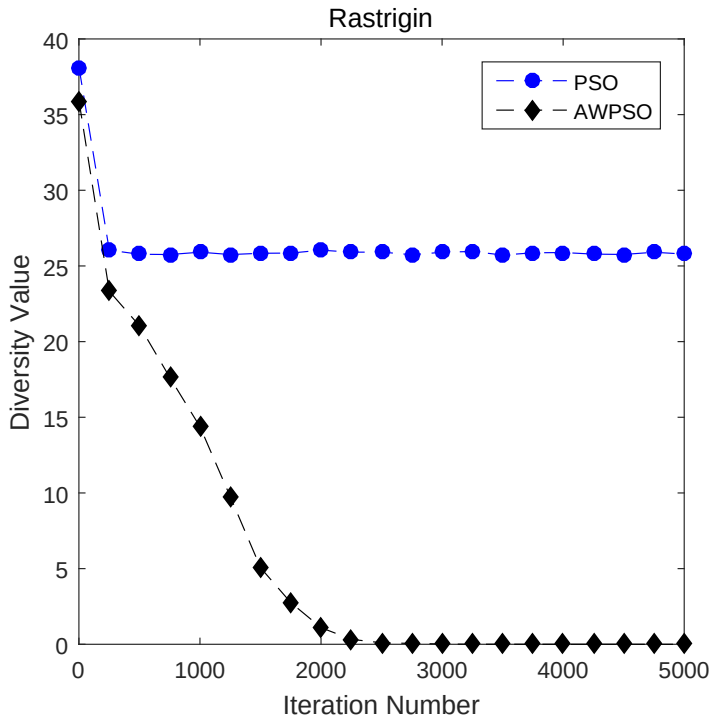

Fig. 3.12 Diversity measure for Rastrigin function $f_3(x)$

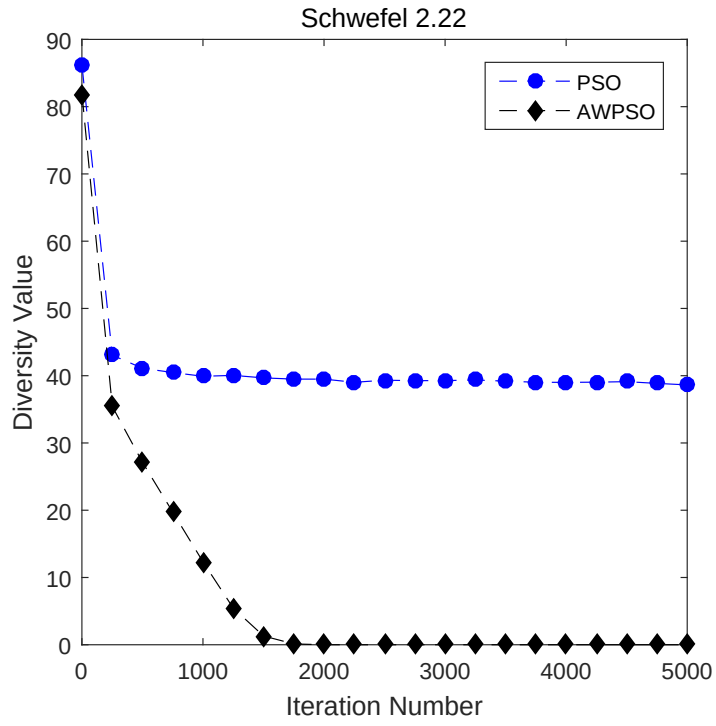

Fig. 3.13 Diversity measure for Schwefel 2.22 function $f_4(x)$

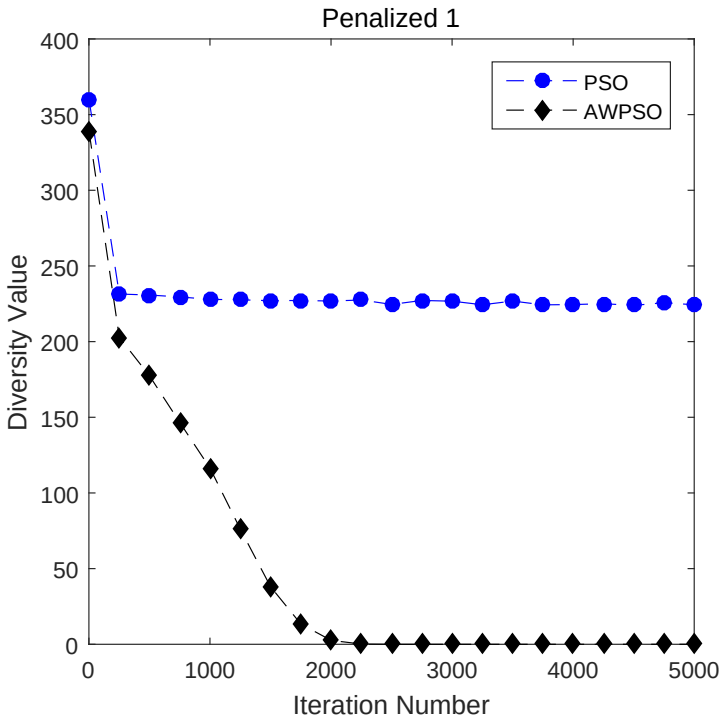

Fig. 3.14 Diversity measure for Penalized 1 function $f_5(x)$

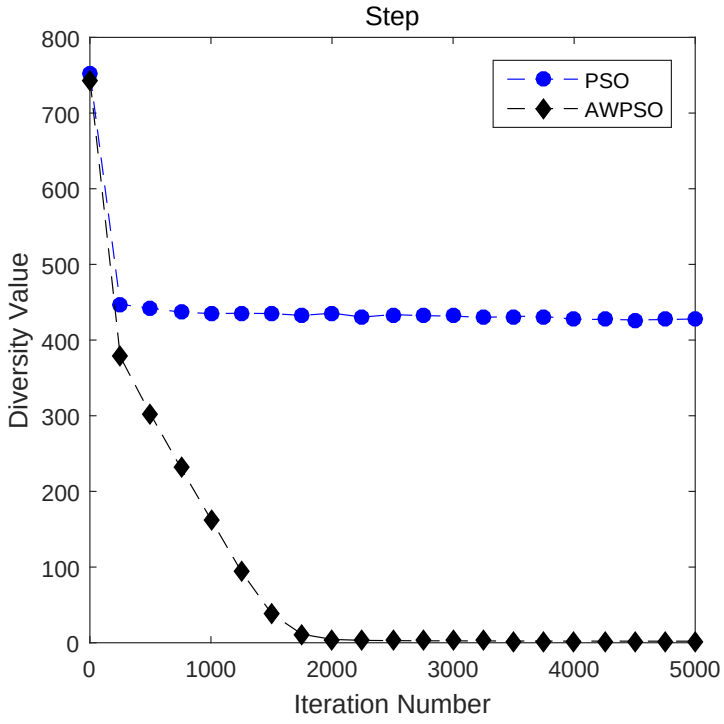

Fig. 3.15 Diversity measure for Step function $f_6(x)$

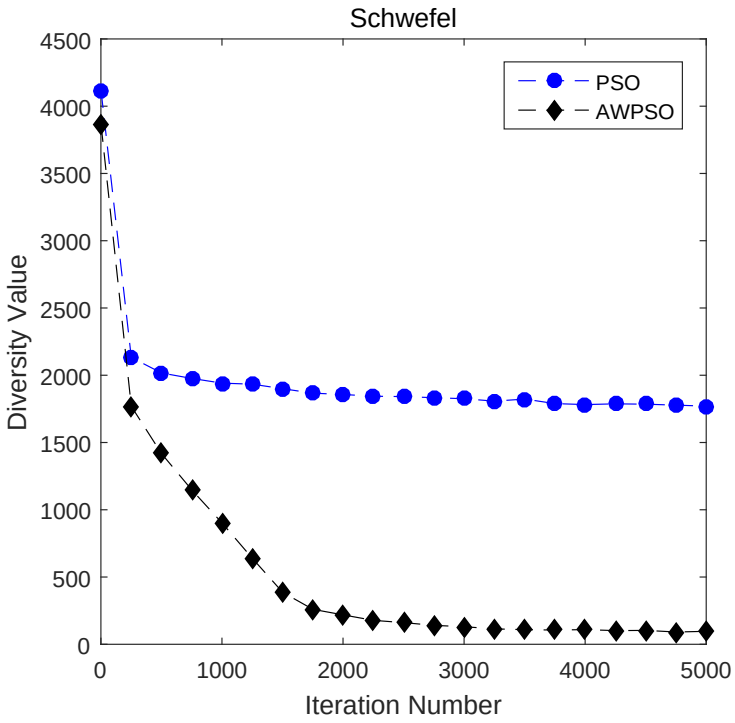

Fig. 3.16 Diversity measure for Schwefel function $f_7(x)$

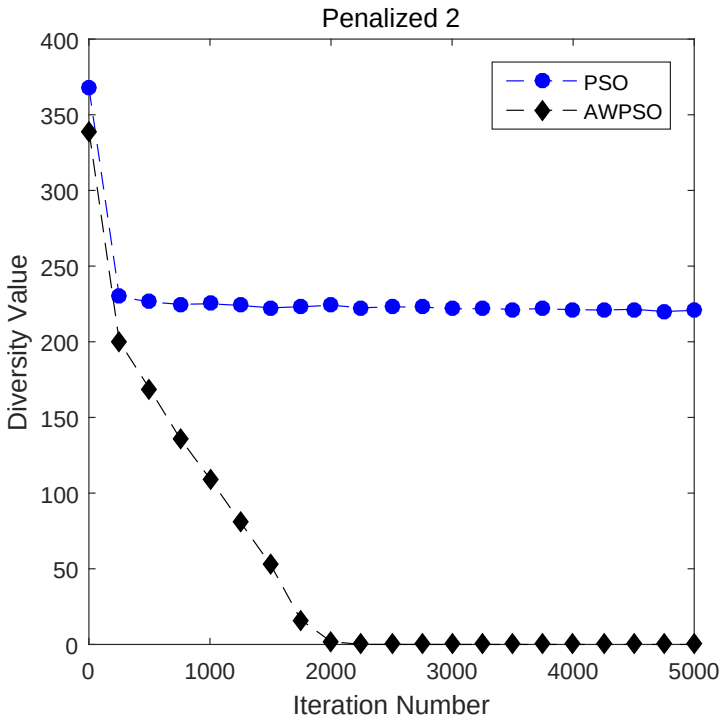

Fig. 3.17 Diversity measure for Penalized 2 function $f_8(x)$

The statistical results of the PSO algorithms are illustrated in Table 3.2. Notably, the minimum, standard deviation and mean fitness value are utilized to evaluate the searching capability of the particle swarm optimizers. The success ratio is used to judge the convergence characteristics, which demonstrates the PSO algorithms' capability of getting rid of the local optima. Notice that all the selected benchmark functions are minimization problems. As such, a smaller fitness value indicates a better solution. In Table 3.2, the proposed AWPSO algorithm obtains smaller minimum fitness value than the classic PSO algorithm, the PSO-LDIW algorithm, the SDPSO algorithm for function (3.5). In addition, the AWPSO algorithm exhibits better performance than the basic PSO algorithm, the PSO-CK algorithm and the SDPSO algorithm for function (3.7). We can see that the minimum fitness value of the AWPSO algorithm is the smallest comparing with all other PSO algorithms for function (3.9) to (3.12). Moreover, the AWPSO algorithm achieves the satisfactory results for most of the benchmark functions by comparing the mean fitness value.

On the other hand, the success ratio is an important criterion to evaluate the evolutionary algorithms. In Table 3.2, only the PSO-LDIW algorithm and the AWPSO algorithm achieve 100% success ratio on function (3.12), which indicates the difficulty of finding the global optimum for the selected benchmark functions. Note that the success ratio of all the benchmark algorithms for Rastrigin function (3.7) and the Schwefel function (3.11) are not satisfactory because these two functions have a large number of local minima, which are hard to find the globally optimal solution, and thus results in a low success ratio. Comparing the success ratio of the PSO algorithms, the AWPSO algorithm demonstrate competitive performance on most of the benchmark functions.

Note that the convergence rate is also a significant performance indicator. In this chapter, the stopping criterion is set as the algorithm finds the globally optimal solution within the threshold. In this case, a smaller number of iteration indicates a better convergence performance of the PSO algorithm. To avoid random phenomena, we repeat the experiment for 50 times on each benchmark function and calculate the mean iteration number. The convergence plot of PSO algorithms is depicted in Fig. 3.18 where the vertical coordinate denotes the number of iteration when the algorithm converges, and the horizontal coordinate represents the number of benchmark function. In Fig. 3.18, we can see that the AWPSO algorithm outperforms the basic PSO algorithm, the PSO-LDIW algorithm and the SDPSO algorithm. The PSO-CK algorithm converges faster than the AWPSO algorithm on function (3.5), function (3.6), function (3.8) and function (3.12). Nevertheless, it is worth mentioning that the overall difference of average convergence rate between the AWPSO algorithm and the PSO-CK algorithm is not large. Importantly, the AWPSO algorithm demonstrates higher success ratio than the PSO-CK algorithm.

The ranking method is employed in our work to evaluate the convergence performance of the PSO algorithms in a quantitative way [66]. The ranking results are shown in Table 3.3. In this work, the smaller the ranking value, the better the convergence performance of the algorithm. It can be seen that the average ranking value of the AWPSO algorithm is the smallest (which is 1.625) among all the PSO algorithms. To summarise, we can arrive at the conclusion that the proposed AWPSO algorithm

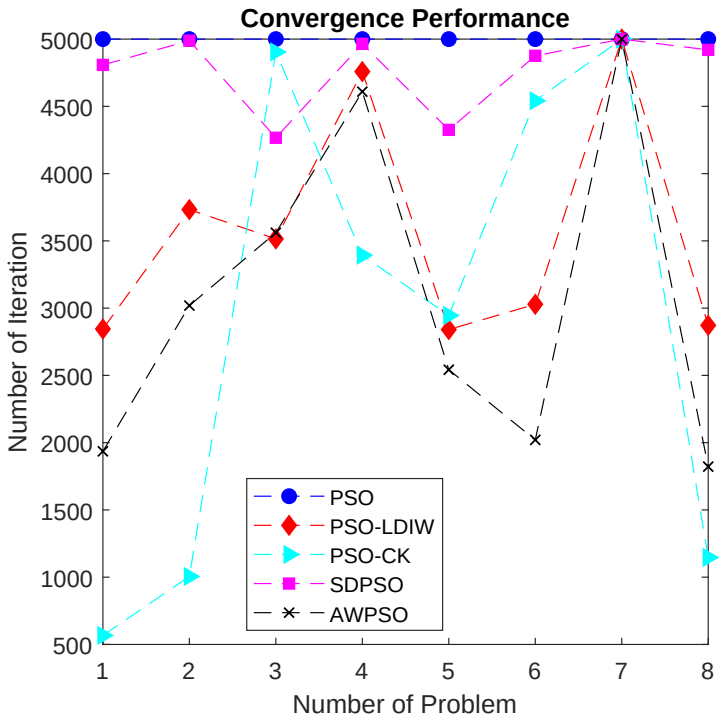

Fig. 3.18 Convergence plot of PSO algorithms

demonstrates competitive performance on the population diversity and the convergence rate.

Table 3.3 Convergence performance evaluation by using the ranking method

		PSO	PSO-LDIW	PSO-CK	SDPSO	AWPSO
$f_1(x)$	Ranking	5	3	1	4	2
$f_2(x)$	Ranking	4	4	1	3	2
$f_3(x)$	Ranking	5	1	4	3	2
$f_4(x)$	Ranking	5	4	1	3	2
$f_5(x)$	Ranking	5	2	3	4	1
$f_6(x)$	Ranking	5	3	1	4	2
$f_7(x)$	Ranking	5	3	1	4	2
$f_8(x)$	Ranking	5	3	1	4	2
	Average Ranking	4.375	2.5	1.875	3.25	1.625

3.4 Conclusion

To conclude, a novel PSO algorithm called the AWPSO algorithm has been proposed in this chapter with hope to improve the convergence rate of the traditional particle swarm optimizer. A sigmoid-function-based adaptive weighting strategy has been introduced where the acceleration coefficients are adaptively controlled by employing a sigmoid function based on the distances from the particle to the global best position and from the particle to its personal best position. The AWPSO algorithm has demonstrated competitive performance on the convergence rate by comparing with four popular PSO algorithms on eight widely used optimization benchmark functions including both unimodal and multimodal cases. In our future research directions, we aim to 1) further improve the AWPSO algorithm in terms of the population diversity, and study the movement behaviors of particles by using the Wilcoxon rank sum test [107, 108, 18]; and 2) apply the AWPSO algorithm to other research fields, such as system engineering and signal processing [157, 185, 20, 125, 200].

Chapter 4

A Novel Randomised Particle Swarm Optimizer

4.1 Motivation

In the past few years, a large number of optimization techniques have been developed by various research communities, e.g., mathematics, telecommunication, computer science [9, 203, 149, 188]. Particularly, the evolutionary computation approaches have proven to be a powerful family of optimization techniques due to their highly efficiency in dealing with global optimization problems [43, 183, 114, 150, 56, 79]. Motivated by the social behaviours of the animal societies, the PSO algorithm has shown to be a powerful family of the EC approaches owing to its strong search ability and relatively fast convergence to the optimal solution [48, 153, 149, 188, 191].

In a PSO algorithm, the acceleration coefficients are crucial parameters in achieving the balance between the global exploration and local searches through the entire problem space [145, 136, 48, 97]. The selection of the acceleration coefficients plays an important role in successfully seeking the globally optimal solution. To improve the search capability of the PSO algorithm, a great many PSO variants have been developed by adjusting the acceleration coefficients, for example, controlling the acceleration coefficients in a time-varying manner [129]. In Chapter 3, the AWPSO algorithm with

a sigmoid-function-based updating strategy is put forward to adaptively adjust the acceleration coefficients in order to enhance the convergence rate of the PSO algorithm.

Intuitively, a properly designed random perturbation (with adequate intensity) on the velocity updating model could lead to allowable variation of the acceleration coefficient that will not affect the convergence of the PSO algorithm but, rather, enhance the population diversity at each iteration, thereby further increasing the possibility of getting rid of local optima. In choosing a candidate for random perturbations, the well-known Gaussian white noise (GWN) appears to be an ideal candidate due to its constant power spectral density and easily tunable intensity at different frequencies when it comes to the implementation [167, 168].

Inspired by above argument of developing randomized algorithms [116], a seemingly natural idea is to introduce the GWNs into the acceleration coefficients of the PSO algorithm with hope to improve the population diversity and alleviate the premature convergence. The advantages of utilizing the GWNs are concluded as twofold: 1) the GWNs in the acceleration coefficients can alter the system dynamics (by means of iterations) which could contribute to a more thorough exploration and exploitation through the problem space; and 2) with the GWNs in place, the particles are entitled to exhibit more complicated dynamical behaviors (than the conventional PSO algorithms) which would enhance the capability of the particles escaping from the local optima and also improve the population diversity.

To conclude the discussions made so far, in this chapter, we endeavor to propose a randomized PSO (RPSO) algorithm where the GWN with adequately adjusted intensity is utilized to randomly perturb the acceleration coefficients in order for the problem space to be explored more thoroughly. The main contributions can be summarized as follows: 1) a novel RPSO algorithm is developed where the GWNs are embedded in the velocity updating model to adjust the acceleration coefficients at each iteration, which helps prevent the undesirable premature convergence; and 2) the proposed RPSO algorithm is comprehensively verified on a series of test functions (including both the unimodal and multimodal cases) and it is demonstrated that the RPSO algorithm outperforms some existing popular variants of PSO algorithms on a series of widely used optimization benchmark functions.

The rest of this chapter is organized as follows. In Section 4.2, the proposed RPSO algorithm is explained in detail. Experiment results, parameter setting and discussions are presented in Section 4.3. Finally, conclusions and future directions are drawn in Section 4.4.

4.2 A New RPSO Algorithm

A novel RPSO algorithm is developed in this section where the GWNs are entered into the updating model for velocity for randomly perturbing (with adequate intensity) the acceleration coefficients with hope to decrease the trapping possibility into the local optima and also seek the optimal solution more thoroughly. The motivation and framework of the proposed RPSO algorithm are illustrated in details.

To control the PSO algorithm in an effective way, the control parameters (such as the inertia weight, the social acceleration coefficient and the cognitive acceleration coefficient) are dominantly crucial during the evolution process. The inertia weight is a significant parameter in controlling the exploration of the search space which is commonly set to be a constant or a dynamically changing value [136]. Similarly, the acceleration coefficients (composed of the cognitive component and the social component) are used to control the movement of the particles towards their personal best position and the global best position discovered by the entire swarm, respectively. In general, the parameter setting of the acceleration coefficients plays an adequate role in achieving the balance between the local search and the global exploration through the optimization process. As such, an appropriate selection of the acceleration coefficients is of vital importance to seek the global optimum effectively and accurately. Unfortunately, some existing PSO algorithms which focus on adjusting the control parameters (such as the PSO-TVAC algorithm) may still easily get trapped in the local optima. Therefore, it is of crucial importance to investigate an advanced parameter selection mechanism to reduce the possibility of trapping into local optima and further enhance the search capability of the PSO algorithms.

We are now ready to introduce our novel RPSO algorithm dedicatedly designed to enhance the search ability of the particles with the hope to thoroughly explore and

exploit the entire problem space. The major novelty of the newly proposed RPSO model is to separately introduce the GWNs into the cognitive acceleration coefficient as well as the social acceleration coefficient to effectively and efficiently seek the optimal solution. By establishing such a new velocity updating model, the RPSO algorithm consists of the following two advantages: 1) the GWNs are separately added to the social and cognitive acceleration coefficients which randomly perturb the movement of the particles at each iteration; and 2) the dynamical behavior of the RPSO algorithm becomes more complicated than the basic PSO algorithm and the particles are therefore allowed to expand their search space, which leads to a more thorough exploration of the problem space with less trapping possibility into the local optima.

4.2.1 Framework of the RPSO Algorithm

For the novel RPSO algorithm, the flowchart is depicted in Fig. 4.1.

The velocity and position of the ith particle are updated based on the following equations:

$$\begin{aligned} v_i(k+1) &= wv_i(k) + r_1(C_p + \delta_1(k))(p_i(k) - x_i(k)) \\ &\quad + r_2(C_g + \delta_2(k))(p_g(k) - x_i(k)) \\ x_i(k+1) &= x_i(k) + v_i(k+1) \end{aligned} \tag{4.1}$$

where k denotes the iteration number; C_p and C_g indicate the acceleration coefficients defined in Eq. (4.3) and Eq. (4.4), respectively; w is inertia weight represented by Eq. (4.2); $\delta_1(k)$ and $\delta_2(k)$ represent two independent GWNs; and r_1 and r_2 are two uniformly distributed random numbers on $[0, 1]$. Notably, the GWNs ($\delta_1(k)$ as well as $\delta_2(k)$) and the random numbers (r_1 and r_2) are mutually independent.

Motivated by the PSO-LDIW [136] and the PSO-TVAC [129] algorithms, the inertia weight w and the acceleration coefficients C_p and C_g of the RPSO algorithm are shown as follows:

$$w = w_{\max} - (w_{\max} - w_{\min}) \times \frac{k}{\text{maxit}} \tag{4.2}$$

$$C_p = (C_{p\max} - C_{p\min}) \times \frac{\text{maxit} - k}{\text{maxit}} + C_{p\min} \tag{4.3}$$

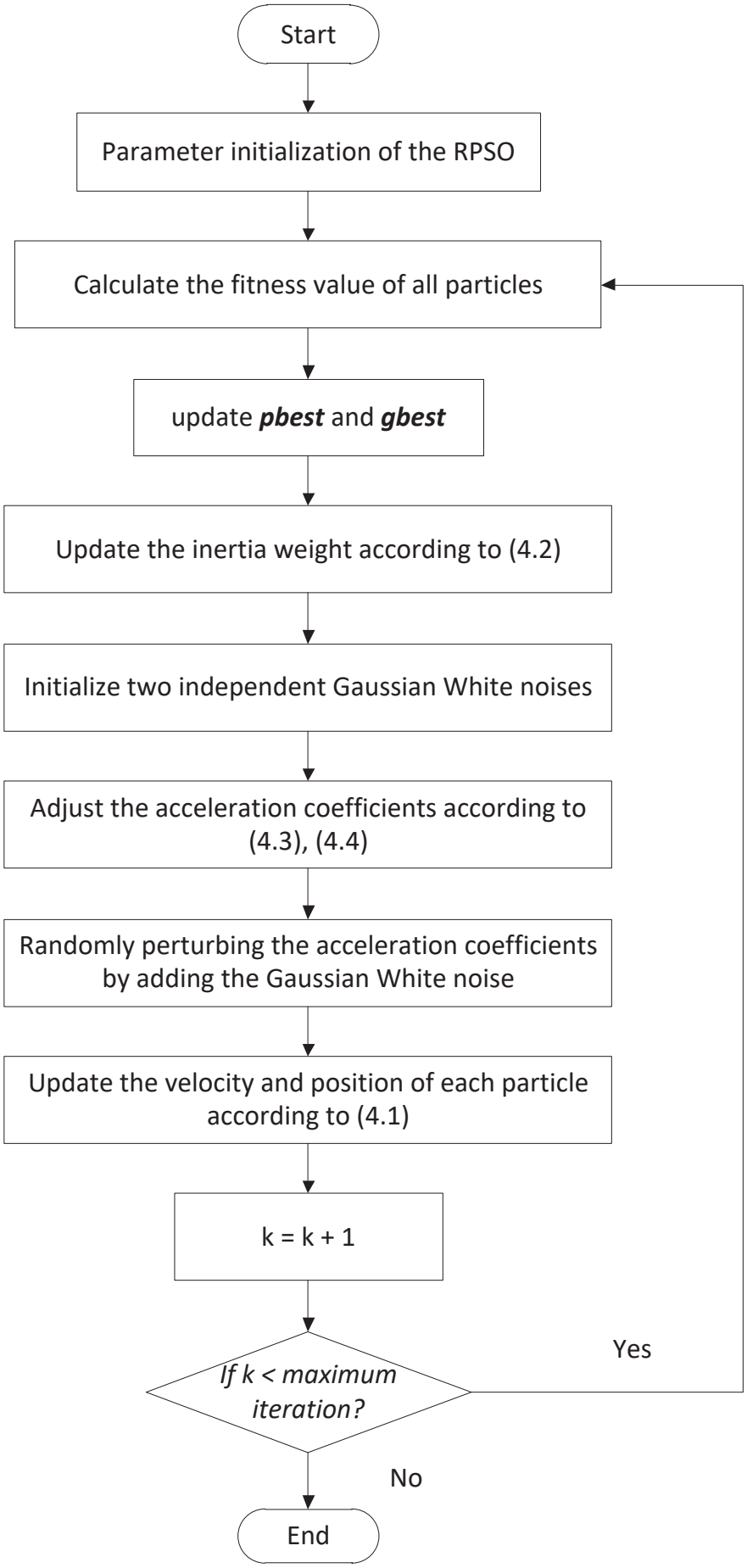

Fig. 4.1 Flowchart of the RPSO algorithm

$$C_g = (C_{g\min} - C_{g\max}) \times \frac{\text{maxit} - k}{\text{maxit}} + C_{g\max} \tag{4.4}$$

where $w_{\max}$, $C_{p\max}$ and $C_{g\max}$ represent the maximum value of the inertia weight w, acceleration coefficients C_p and C_g, respectively; $w_{\min}$, $C_{p\min}$ and $C_{g\min}$ denote the minimum value of w, C_p and C_g, respectively; and $maxit$ indicates the maximum iteration.

4.3 Results and Discussion

In this section, the performance of the proposed RPSO algorithm is evaluated and discussed by comparing with some widely used PSO algorithms. In the simulation, some popular test functions including both of the unimodal and multimodal cases are taken into account to provide a comprehensive review of the optimization performance of the developed RPSO algorithm. In this chapter, all the experiments are implemented by using MATLAB 2017b on a PC with the Intel Core $i5-4590$ 3.30 GHz CPU and the Microsoft Windows 7 Enterprise 64-bit operating system.

In our simulation, the swarm size is set as $S = 30$ and the dimension of the problem space is $D = 30$. Furthermore, the maximum number of iteration is $k = 10000$ for all the tested PSO algorithms. To strengthen the reliability of the simulation results, each experiment is repeated independently for 50 times. The parameters of the compared PSO algorithms are set up according to the literature [82, 135, 129, 27, 149, 188]. For the RPSO algorithm, the inertia weight w is linearly decreased from 0.9 to 0.4. The acceleration coefficients C_p and C_g are belonging to $[0.5, 2.5]$ where $C_{p\max} = C_{g\max} = 2.5$ and $C_{p\min} = C_{g\min} = 0.5$. The mean value and the variance of the GWNs $\delta_1(k)$ and $\delta_2(k)$ are both 0 and 0.07, respectively.

4.3.1 Test Functions

In this chapter, eight well-known test functions are selected for evaluating the search ability of the proposed RPSO algorithm by comparing with six popular PSO algorithms. The selected PSO algorithms include the standard PSO algorithm [82], the PSO-LDIW algorithm [135], the PSO-TVAC algorithm [129], the PSO-CK algorithm [27], the

SPSO algorithm [149], and the SDPSO algorithm [188]. Among the selected test functions, $f_1(x)$ (the Sphere function) is a typical unimodal function which is often utilized to justify the convergence rate of the EC approaches; $f_2(x)$ (the Rosenbrock function), referred to as the Rosenbrock's banana function, is a widely-used test problem for optimization algorithms; $f_3(x)$ (the Rastrigin function) and $f_5(x)$ (the Griewank function) have a large number of local optima, which are hard to discover the globally optimal solution; and other selected test functions are also popular benchmark functions. It should be pointed out that all the test functions are minimization problems and all of them have a global minimum. Let $x = (x_1, x_2, \cdots, x_D)$ where $D = 30$ is the dimension of the problem space. The mathematical formulations of the test functions are given as below.

$$\text{Sphere}: f_1(x) = \sum_{i=1}^{D} x_i^2. \tag{4.5}$$

$$\text{Rosenbrock}: f_2(x) = \sum_{i=1}^{D-1} (100(x_{i+1} - x_i)^2 + (x_i - 1)^2). \tag{4.6}$$

$$\text{Rastrigin}: f_3(x) = \sum_{i=1}^{D} (x_i^2 - 10\cos 2\pi x_i + 10). \tag{4.7}$$

$$\text{Schwefel 1.2}: f_4(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2. \tag{4.8}$$

$$\text{Griewank}: f_5(x) = 1 + \frac{1}{4000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}). \tag{4.9}$$

$$\begin{aligned} \text{Penalized 1}: f_6(x) = \frac{\pi}{D} \Big(& 10\sin^2(\pi y_1) \\ & + \sum_{i=1}^{D-1} (y_i - 1)^2 \Big(1 + 10\sin^2(\pi y_{i+1})\Big) \\ & + (y_D - 1)^2 \Big) + \sum_{i=1}^{D} u(x_i). \end{aligned} \tag{4.10}$$

$$y_i = 1 + 1/4(x_i + 1),$$

$$u(x_i) = \begin{cases} 100(-x_i - 10)^4, & x_i < -10, \\ 0, & |x_i| \leq 10, \\ 100(x_i - 10)^4, & x_i > 10. \end{cases}$$

$$\text{Step}: f_7(x) = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2. \tag{4.11}$$

$$\begin{aligned} \text{Penalized 2}: f_8(x) = 0.1\Big(& \sin^2(3\pi x_1) \\ & + \sum_{i=1}^{D-1} (x_i - 1)^2 \Big(1 + \sin^2(3\pi x_{i+1})\Big) \\ & + (x_D - 1)^2 \Big(1 + \sin^2(2\pi x_D)\Big)\Big) \\ & + \sum_{i=1}^{D} u(x_i). \end{aligned} \tag{4.12}$$

$$u(x_i) = \begin{cases} 100(-x_i - 5)^4, & x_i < -5, \\ 0, & |x_i| \leq 5, \\ 100(x_i - 5)^4, & x_i > 5. \end{cases}$$

The detailed information of the test functions is shown in Table 4.1 including the name of the test functions, the search range of each optimization problem, the maximum velocity of the particle for each test function, the threshold, and minimum of the test functions. Notably, the search range of the test function which indicates the range of the search space is determined by the literature [176]. Furthermore, the maximum velocity of each particle in PSO approaches is usually limited by a certain value with the hope to avoid searching outside the predefined search range. Due to empirical investigations on the test functions, the maximum velocity is often set up by $10 - 20\%$ of the dynamic range of each dimension for different test functions [129, 48]. In our simulation, the maximum velocity is set to be 20% of the dynamical range.

Table 4.1 Test function configuration

Test Functions	Dimension	Search Range	Maximum Velocity	Threshold	Minimum
$f_1(x)$: Sphere	30	$[-100, 100]$	40	0.01	0
$f_2(x)$: Rosenbrock	30	$[-30, 30]$	12	100	0
$f_3(x)$: Rastrigin	30	$[-5.12, 5.12]$	2.048	50	0
$f_4(x)$: Schwefel 1.2	30	$[-100, 100]$	40	0.01	0
$f_5(x)$: Griewank	30	$[-600, 600]$	240	0.01	0
$f_6(x)$: Penalized 1	30	$[-50, 50]$	20	0.01	0
$f_7(x)$: Step	30	$[-100, 100]$	40	0.01	0
$f_8(x)$: Penalized 2	30	$[-50, 50]$	20	0.01	0

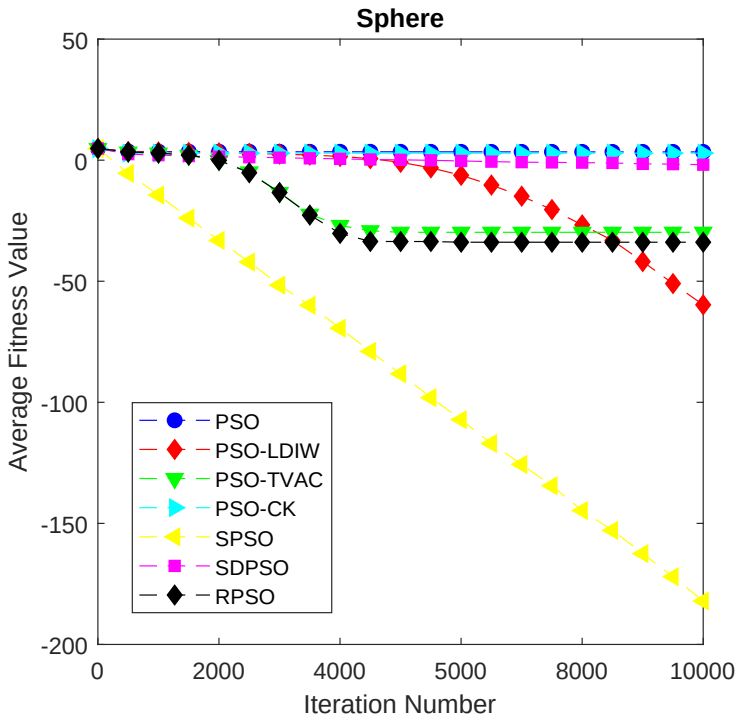

Fig. 4.2 Algorithm Convergence Characteristics for Sphere function $f_1(x)$

4.3.2 Experimental Studies of the RPSO Algorithm

To evaluate the solution quality of the proposed RPSO algorithm, three popular performance indicators (including the convergence rate, success ratio and population diversity) are utilized. Note that the success ratio is an important criterion to measure the accuracy of the PSO approaches. The population diversity is used to evaluate the solution quality. In this chapter, the convergence plots of the PSO algorithms are depicted from Figs. 4.2-4.9 to demonstrate the convergence rate of the PSO algorithms where the vertical coordinate indicates the logarithm value of the average fitness value and the horizontal coordinate denotes the number of iteration. Furthermore, the minimum, mean and standard deviation fitness values of the PSO algorithms are employed to demonstrate the solution quality of the adopted PSO approaches in Table 4.2.

From the figures, it is clear that the proposed RPSO algorithm exhibits competitive performance than other selected PSO variants. In Figs. 4.3-4.5 and Figs. 4.7-4.8, the RPSO algorithm obtains the best average fitness value among the selected PSO

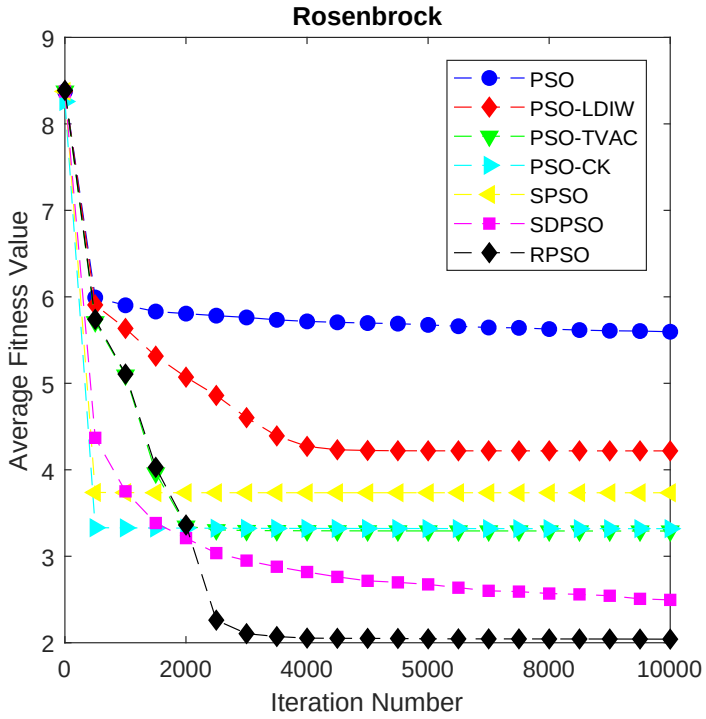

Fig. 4.3 Algorithm Convergence Characteristics for Ackley function $f_2(x)$

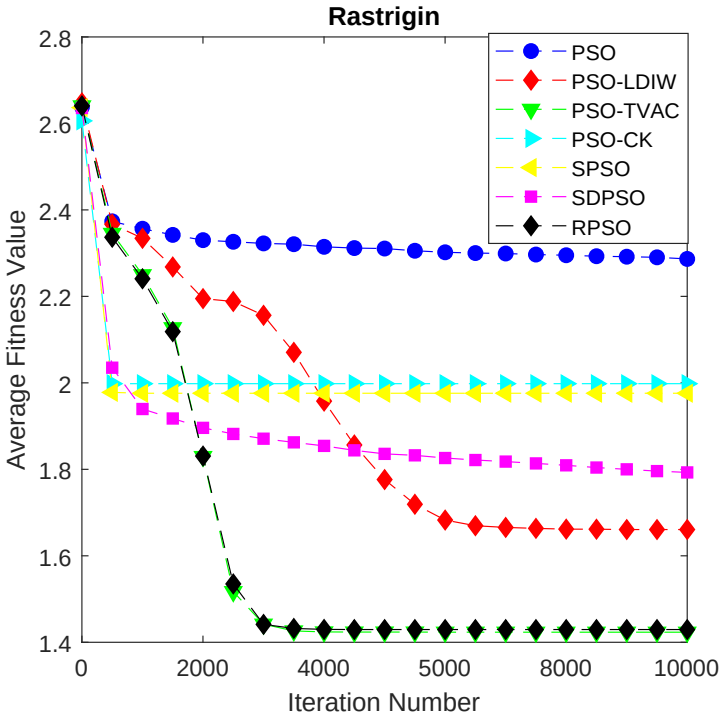

Fig. 4.4 Algorithm Convergence Characteristics for Rastrigin function $f_3(x)$

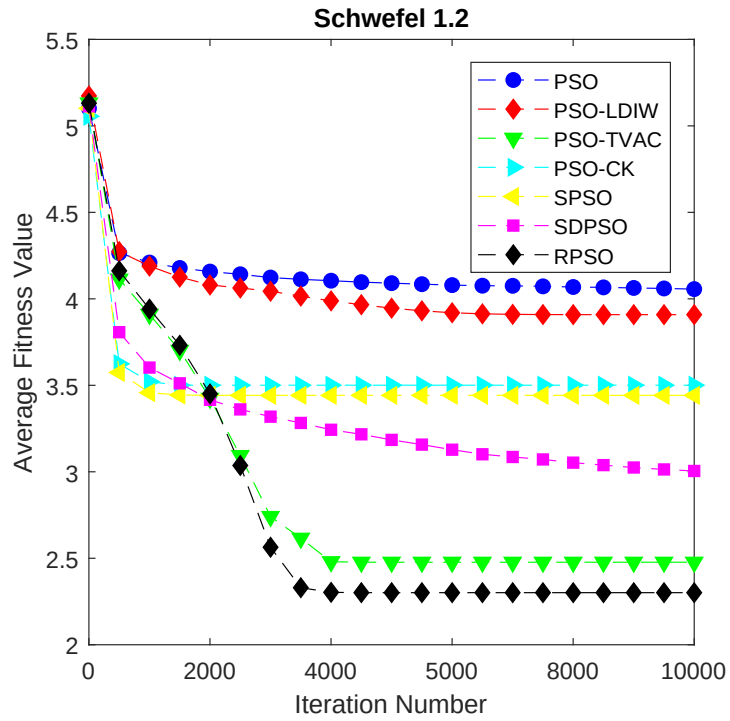

Fig. 4.5 Algorithm Convergence Characteristics for Schwefel 2.22 function $f_4(x)$

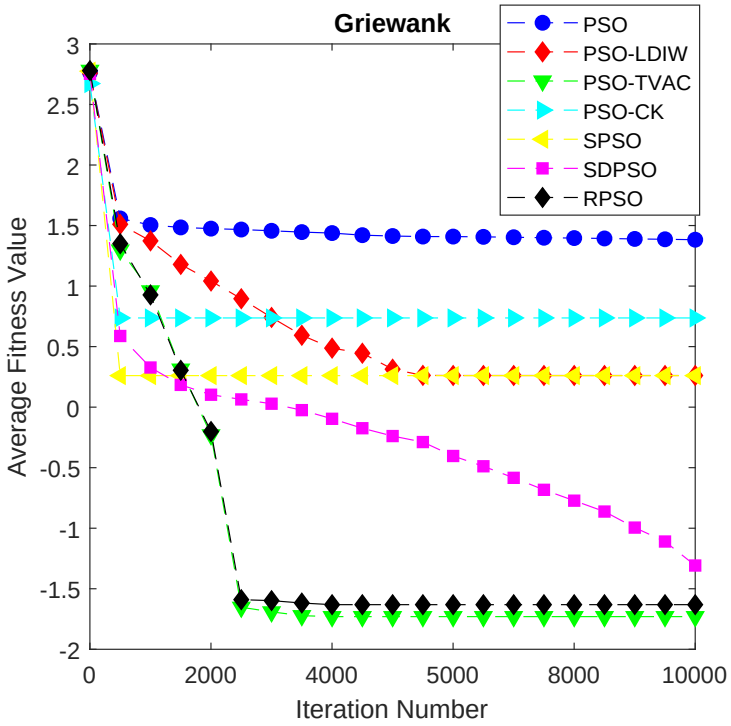

Fig. 4.6 Algorithm Convergence Characteristics for Griewank function $f_5(x)$

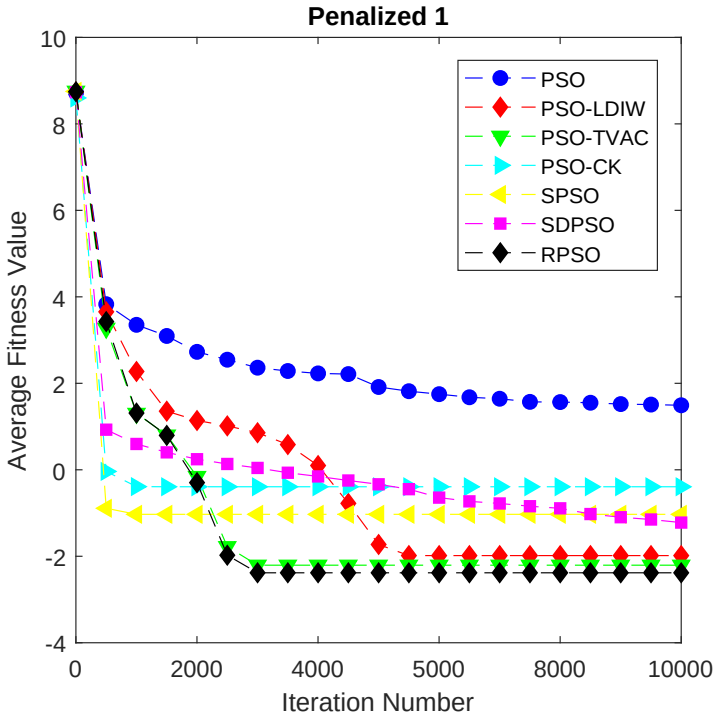

Fig. 4.7 Algorithm Convergence Characteristics for Penalized 1 function $f_6(x)$

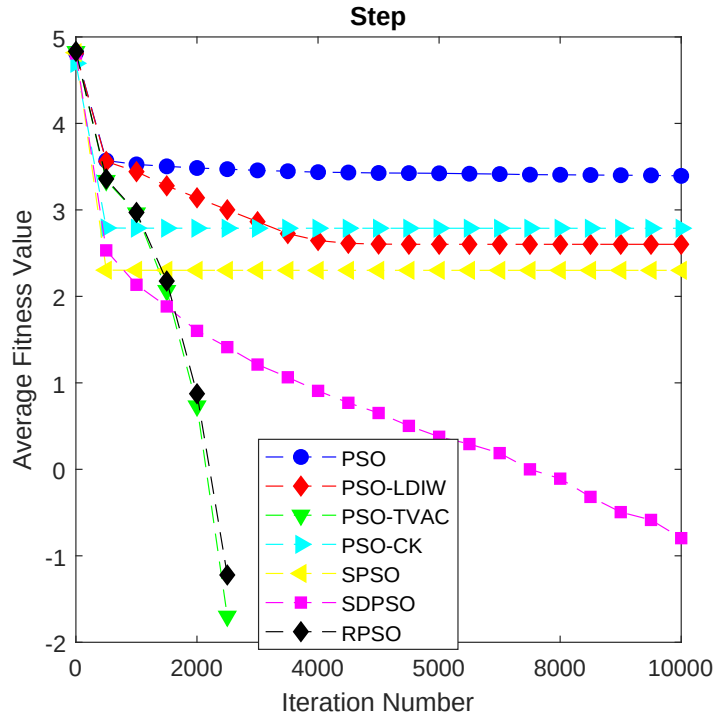

Fig. 4.8 Algorithm Convergence Characteristics for Step function $f_7(x)$

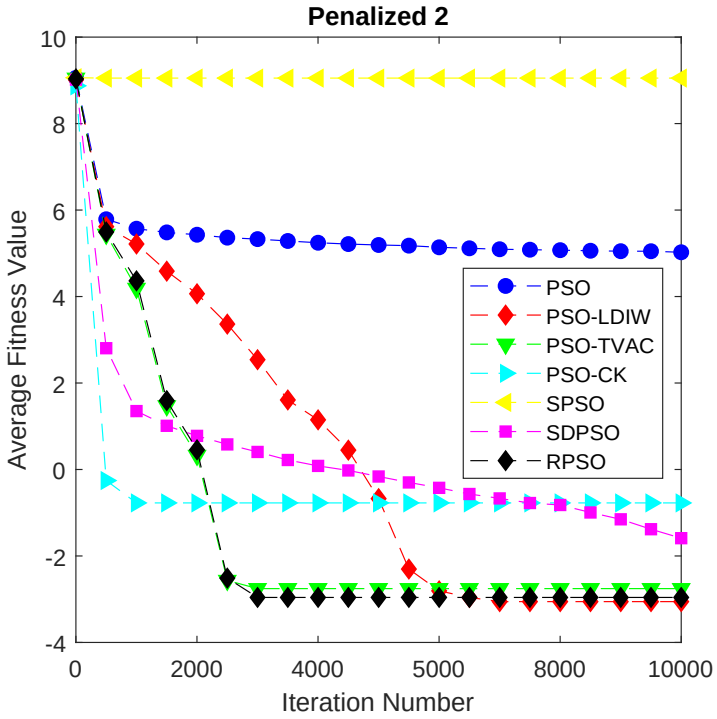

Fig. 4.9 Algorithm Convergence Characteristics for Penalized 2 function $f_8(x)$

algorithms, and the convergence rate is satisfactory. Additionally, the RPSO algorithm discovers the globally optimal solution of function (4.11). Although the convergence characteristics of the RPSO algorithm is not the best among all the PSO variants for function (4.5), function (4.9) and function (4.12), the difference between the best minimum value is not significant.

The ranking method is employed to quantitatively evaluate the convergence performance of the PSO algorithms. The ranking results are displayed in Table 4.3. It can be seen that the mean ranking value of the RPSO algorithm is 1.5, which is the smallest one among all the selected PSO algorithms. Overall, the convergence behavior of the RPSO algorithm outperforms other selected PSO algorithms.

The statistical results of the performance evaluation of the selected PSO approaches are shown in Table 4.2. By analyzing the experiment results of the RPSO algorithm on the chosen 30-D optimization problems, the proposed RPSO algorithm outperforms six popular PSO algorithms in terms of both of the success ratio and the population diversity. Although it is tough to find the optimal solution, the algorithm meets the

Table 4.2 Statistical Results of PSO Algorithms

		PSO	PSO-LDIW	PSO-TVAC	PSO-CK	SPSO	SDPSO	RPSO
$f_1(x)$	Minimum	1.44×10^{3}	2.38×10^{-64}	3.97×10^{-57}	6.07×10^{-175}	3.23×10^{-196}	3.42×10^{-5}	1.78×10^{-55}
	Mean	2.55×10^{3}	1.81×10^{-60}	1.62×10^{-30}	8.00×10^{2}	8.58×10^{-183}	3.59×10^{-67}	1.30×10^{-34}
	Std. Dev.	1.45×10^{3}	8.82×10^{-60}	9.45×10^{-30}	2.74×10^{3}	0.0000	2.88×10^{-2}	5.61×10^{-34}
	Ratio	0%	100%	100%	92%	100%	74%	100%
$f_2(x)$	Minimum	2.28×10^{5}	5.75×10^{-3}	2.31×10^{-3}	7.67×10^{-6}	5.33×10^{-4}	2.77×10^{1}	3.82×10^{-4}
	Mean	3.95×10^{5}	1.65×10^{4}	1.96×10^{3}	2.09×10^{3}	5.43×10^{3}	3.13×10^{2}	1.10×10^{2}
	Std. Dev.	1.12×10^{5}	3.48×10^{4}	1.27×10^{4}	1.27×10^{4}	2.16×10^{4}	6.95×10^{2}	4.32×10^{2}
	Ratio	0%	64%	90%	80%	92%	54%	94%
$f_3(x)$	Minimum	1.34×10^{2}	1.29×10^{1}	1.39×10^{1}	4.97×10^{1}	3.18×10^{1}	2.69×10^{1}	1.29×10^{1}
	Mean	1.93×10^{2}	4.58×10^{1}	2.65×10^{1}	9.96×10^{1}	9.46×10^{1}	6.21×10^{1}	2.69×10^{1}
	Std. Dev.	2.22×10^{1}	2.18×10^{1}	6.5499	2.96×10^{1}	3.16×10^{1}	2.15×10^{1}	1.00×10^{1}
	Ratio	0%	60%	100%	2%	8%	36%	96%
$f_4(x)$	Minimum	4.05×10^{3}	9.74×10^{-4}	1.71×10^{-8}	2.87×10^{-24}	5.79×10^{-14}	8.60×10^{1}	1.13×10^{-9}
	Mean	1.14×10^{4}	8.10×10^{3}	3.00×10^{2}	3.17×10^{3}	2.77×10^{3}	1.01×10^{3}	2.00×10^{2}
	Std. Dev.	5.67×10^{3}	7.93×10^{3}	1.20×10^{3}	3.94×10^{3}	4.47×10^{3}	1.81×10^{3}	9.90×10^{2}
	Ratio	0%	10%	92%	54%	62%	0%	92%
$f_5(x)$	Minimum	1.45×10^{1}	0.0000	0.0000	0.0000	0.0000	1.83×10^{-5}	0.0000
	Mean	2.42×10^{1}	1.8255	1.86×10^{-2}	5.4554	1.8216	4.92×10^{-2}	2.34×10^{-2}
	Std. Dev.	1.32×10^{1}	1.28×10^{1}	2.16×10^{-2}	2.17×10^{1}	1.28×10^{1}	7.86×10^{-2}	2.25×10^{-2}
	Ratio	0%	36%	54%	30%	56%	18%	38%
$f_6(x)$	Minimum	1.79×10^{1}	1.57×10^{-32}	1.57×10^{-32}	1.57×10^{-32}	1.57×10^{-32}	5.07×10^{-5}	1.57×10^{-32}
	Mean	3.11×10^{1}	1.04×10^{-2}	6.22×10^{-3}	4.05×10^{-1}	9.33×10^{-2}	6.01×10^{-2}	4.15×10^{-3}
	Std. Dev.	1.12×10^{1}	3.14×10^{-2}	2.49×10^{-2}	6.66×10^{-1}	1.44×10^{-1}	1.88×10^{-1}	2.05×10^{-2}
	Ratio	0%	90%	94%	40%	54%	72%	96%
$f_7(x)$	Minimum	1.48×10^{3}	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	Mean	2.47×10^{3}	4.00×10^{2}	0.0000	6.13×10^{2}	2.00×10^{2}	1.60×10^{-1}	0.0000
	Std. Dev.	1.41×10^{3}	1.98×10^{3}	0.0000	2.40×10^{3}	1.41×10^{3}	5.48×10^{-1}	0.0000
	Ratio	0%	96%	100%	24%	92%	90%	100%
$f_8(x)$	Minimum	1.49×10^{4}	1.35×10^{-32}	1.35×10^{-32}	1.35×10^{-32}	5.83×10^{-31}	2.11×10^{-5}	1.35×10^{-32}
	Mean	1.06×10^{5}	8.79×10^{-4}	1.76×10^{-3}	1.69×10^{-1}	1.13×10^{9}	2.58×10^{-2}	1.10×10^{-3}
	Std. Dev.	5.98×10^{4}	3.01×10^{-3}	4.07×10^{-3}	7.10×10^{-1}	3.00×10^{8}	3.78×10^{-2}	3.33×10^{-3}
	Ratio	0%	92%	84%	50%	2%	48%	90%

requirement if the fitness value reaches the specific threshold of each test function. The RPSO algorithm obtains best success ratio in function (4.5), function (4.6), function (4.8), function (4.10) and function (4.11). For function (4.7), function (4.9) and function (4.12), the RPSO algorithm also exhibits competitive success ratio than other PSO algorithms. Notably, the success ratio of each PSO algorithms on function (4.9) is very low because the Griewank function consists of a large number of local minima, which is difficult to discover the globally optimal solution. Overall, we can draw the conclusion that the RPSO algorithm is capable of escaping from the local optima.

In Table 4.2, the mean, minimum, and standard deviation of the fitness values are also presented. Note that the minimum fitness value represents the optimal solution found by the PSO algorithm. It should be mentioned that all the test functions utilized in this chapter are minimization problems. In this regard, a smaller fitness value indicates a better solution explored by the PSO algorithm. The RPSO algorithm

Table 4.3 Convergence performance evaluation by using the ranking method

	PSO	PSO-LDIW	PSO-TVAC	PSO-CK	SPSO	SDPSO	RPSO
$f_1(x)$ Ranking	7	2	4	6	1	5	3
$f_2(x)$ Ranking	7	6	3	4	5	2	1
$f_3(x)$ Ranking	7	3	2	6	5	4	1
$f_4(x)$ Ranking	7	6	2	5	4	3	1
$f_5(x)$ Ranking	7	4	1	6	4	3	2
$f_6(x)$ Ranking	7	3	2	6	5	4	1
$f_7(x)$ Ranking	7	5	2	6	4	3	1
$f_8(x)$ Ranking	6	1	3	5	7	4	2
Average Ranking	6.875	3.75	2.375	5.5	4.375	3.5	1.5

obtains the smallest minimum fitness value on function (4.7) and functions (4.9)-(4.12). In addition, the RPSO algorithm obtains satisfactory minimum fitness values on the rest of the test functions. It is worth mentioning that the RPSO algorithm also gets the smallest mean and standard deviation of the fitness value on function (4.6), function (4.8), and functions (4.10)-(4.12), which indicates that the population diversity of the RPSO algorithm is competitive than other compared PSO algorithms. Furthermore, the RPSO algorithm outperforms the standard PSO algorithm, the PSO-LDIW algorithm, the PSO-CK algorithm, the SPSO algorithm and the SDPSO algorithm on the mean, minimum, and standard deviation of the fitness values as well as the success ratio for function (4.7). For function (4.11), the success ratio of the RPSO algorithm is 100% which indicates that the globally optimal solution is discovered for all the repeats. In this case, we can draw the conclusion that the search ability of the RPSO algorithm is competitive than the compared PSO algorithms. To summarize, the proposed RPSO algorithm demonstrates superior performance over the compared popular PSO algorithms in terms of the strong ability to escape from the local optima, the satisfactory convergence performance and population diversity.

4.4 Conclusion

In this chapter, a RPSO algorithm has been proposed to improve the search ability of the basic PSO approach. The GWNs have been added to the social acceleration

coefficient and the cognitive acceleration coefficient separately to randomly change the acceleration coefficients. Experimental results have shown that the introduced RPSO algorithm outperforms six existing PSO algorithms on eight popular test functions. In the near future, we will 1) further investigate advanced approaches to improving the population diversity of the RPSO algorithm for large-scale and complex optimization problems; 2) apply the proposed RPSO algorithm to other research areas, e.g., system science and telecommunication [167, 168]; and 3) extend our work to multi-objective optimization [122, 95, 202, 147, 205].

Chapter 5

A Randomly Occurring Distributedly Delayed Particle Swarm Optimizer

5.1 Motivation

Owing to its wide application potential and satisfactory performance, the past few decades have witnessed the rapid development of the PSO algorithm [136, 46, 192, 193, 6]. It is worth mentioning that the PSO algorithm suffered from the premature convergence problem especially for high-dimensional optimization problems. Therefore, it is of vital importance to further improve the search capability of the PSO algorithm. Note that we develop the AWPSO algorithm in Chapter 3 and the RPSO algorithm in Chapter 4 by modifying the control parameters to improve the convergence and the population diversity. To further improve the PSO algorithm in terms of the updating behavior of the particles, we aim to put forward a new variant PSO algorithm by designing new topological structures.

It should be pointed out that the PSO algorithms perform well by adding certain time-delays in the velocity updating model, see [149, 140, 188]. In the existing delayed PSO algorithms, the time-delay terms (composed of both personal and global best particles in the velocity updating model) contribute significantly to the full use of

historical information and the thorough exploration of the search space, by which the convergence behaviors of PSO algorithms are improved and the capability of getting rid of local optima is enhanced. Time-delay is a physical phenomenon existing in dynamical systems such as single-frequency global positioning systems [51] and genetic regulatory networks [149]. According to the way they occur, time delays can be categorized as constant, time-varying, discrete and distributed ones, see e.g. [165, 139].

Distributed time-delays exhibit a distinct spatial nature that models delay in signal propagations distributed through an amount of parallel channels/pathways during a certain time period. So far, the dynamical behaviors of complex systems (e.g. neural networks [165, 139]) with distributed time-delays have been well studied. Intuitively, a natural idea is to introduce certain distributed time-delays in the PSO algorithm with the hope to enhance the capability of escaping from the local optima and getting rid of the problem of premature convergence. As compared with the discrete time-delays in [149, 140, 188], distributed time-delays could have the following two advantages: 1) a better use of longer (more accumulated) history of the population evolution leading to a better accuracy; and 2) a more complicated dynamical behavior leading to less possibility of trapping local optima. Furthermore, to play an adequate tradeoff between the convergence and the diversity, the introduced distributed time-delays could be made randomly occurring with reasonably small probability. As such, the main purpose of this chapter is to launch a major study on a novel randomly occurring distributedly delayed PSO (RODDPSO) algorithm with applications in healthcare informatics.

Motivated by the above discussions, the purpose of this chapter is to propose a RODDPSO algorithm. The main contributions of this chapter can be summarized as follows: *A novel RODDPSO algorithm is introduced where the randomly occurring distributed time-delay terms not only contribute to a) a thorough exploration of the entire search space; b) a significant reduction of the possibility of trapping local optima; and c) a proper balance between the local and global search abilities.*

The remaining part of this chapter is organized as follows. A novel RODDPSO algorithm is introduced in Section 5.2. Experiment results of the RODDPSO algorithm are presented in Section 5.3. Finally, conclusions and discussions on relevant future work are presented in Section 5.4.

5.2 A Novel RODDPSO Algorithm

In this section, a novel RODDPSO algorithm is proposed to further improve the search ability of the traditional PSO algorithm. The main novelty of the proposed RODDPSO lies in the introduction of the randomly occurring distributed time-delays into the velocity updating model. More specifically, a certain number of historical personal best particles and global best particles are randomly selected according to the evolutionary state. Note that the delayed terms are selected by multiplying a random number which is 0 or 1. Compared to the traditional delayed PSO algorithms, the newly introduced randomly occurring distributed time-delays in the velocity updating model make it possible for us to 1) make better use of accumulated history about the population evolution with better accuracy; 2) pursue stronger capability of avoiding local optima trapping problems; and 3) keep an adequate balance between the convergence and the diversity. As such, the proposed RODDPSO could explore and exploit the search space more thoroughly than the traditional PSO algorithm.

5.2.1 Framework of the RODDPSO Algorithm

The velocity and position in the novel RODDPSO algorithm are updated as follows:

$$\begin{aligned} v_i(k+1) = {} & wv_i(k) + c_1r_1(p_i(k) - x_i(k)) + c_2r_2(p_g(k) - x_i(k)) \\ & + m_l(\xi)c_3r_3\sum_{\tau=1}^{N}\alpha_{(\tau)}(p_i(k-\tau) - x_i(k)) \\ & + m_g(\xi)c_4r_4\sum_{\tau=1}^{N}\alpha_{(\tau)}(p_g(k-\tau) - x_i(k)), \\ x_i(k+1) = {} & x_i(k) + v_i(k+1), \end{aligned} \tag{5.1}$$

where k denotes the current iteration number; w is the inertia weight defined in equation (2.2); acceleration coefficients c_1 and c_2 are updated according to equations (2.3) and (2.4), respectively; c_3 and c_4 are the acceleration coefficients for distributed time-delay terms, which are equal to c_1 and c_2, i.e., $c_1 = c_3$ and $c_2 = c_4$; N represents the upper bound of the distributed time-delays; $\alpha_{(\tau)}$ declares a N-dimensional vector where each element is randomly chosen from 0 or 1; $r_i(i = 1, 2, 3, 4)$ are random numbers which

are uniformly distributed in $[0, 1]$; $m_l(\xi)$ and $m_g(\xi)$ represent the intensity factors of the distributed time-delay terms according to the evolutionary state ξ.

It is worth mentioning the relationship between the delayed iteration number τ and the current iteration number k. Note that the velocity updating model performs according to (5.1) when τ is smaller than k, and otherwise we set $\tau = 0$. On the other hand, the selections of the inertia weight and acceleration coefficients are very important in implementing PSO algorithms. The balance of the local and global searching performance is obtained by adjusting the inertia weight. In this chapter, the selection of inertia weight adopts the linearly decreasing strategy proposed in [136] with equation (2.2). Due to the success in improving the search ability of conventional PSO algorithms by employing time-varying acceleration coefficients in [129], we adopt the time-varying strategy to adjust acceleration coefficients with equations (2.3) and (2.4).

The flowchart of the novel RODDPSO algorithm is given in Fig. 5.1.

5.2.2 Evolutionary State

In the proposed RODDPSO algorithm, the velocity and position equations are updated according to the evolutionary state depending on the evolutionary factor as mentioned in [192, 149]. The searching characteristics of the PSO algorithm are revealed through the four evolutionary states, i.e., the convergence state, the exploitation state, the exploration state, and the jumping-out state denoted by $\xi(k) = 1$, $\xi(k) = 2$, $\xi(k) = 3$ and $\xi(k) = 4$, respectively.

As mentioned in [192], the evolutionary factor is calculated based on the distance between the particles. The mean distance between the ith particle and other particles denoted by d_i is given as follows:

$$d_i = \frac{1}{S-1} \sum_{j=1, j\neq i}^{S} \sqrt{\sum_{k=1}^{D} (x_{ik} - x_{jk})^2}, \tag{5.2}$$

where S denotes the swarm size and D represents the dimension of the particle. The evolutionary factor denoted by E_f is shown as follows:

$$E_f = \frac{d_g - d_{\min}}{d_{\max} - d_{\min}}, \tag{5.3}$$

where d_g represents the global best particle among d_i; $d_{\min}$ and $d_{\max}$ represent the minimum and maximum of d_i in the swarm, respectively.

In this chapter, the equal division strategy is employed to classify the four evolutionary states represented by $\xi(k)$ as follows:

$$\xi(k) = \begin{cases} 1, & 0.00 \leq E_f < 0.25, \\ 2, & 0.25 \leq E_f < 0.50, \\ 3, & 0.50 \leq E_f < 0.75, \\ 4, & 0.75 \leq E_f \leq 1.00. \end{cases} \tag{5.4}$$

where $\xi(k) = 1, 2, 3, 4$ represent the convergence state, the exploitation state, the exploration state, and the jumping-out state, respectively. Detailed information about the four evolutionary states can be found in the literature [192, 149, 140, 188].

5.2.3 Velocity Updating Strategy Based on Randomly Occurring Distributed Time-delay

In this chapter, a novel velocity updating strategy with randomly occurring distributed time-delays is demonstrated for four aforementioned evolutionary states as below:

- In the convergence state denoted by $\xi(k) = 1$, the particles are trying to fly into the globally optimal region as soon as possible. Therefore, the velocity updating model in the traditional PSO algorithm is employed, and the distributed time-delay terms are ignored by setting the intensity factor to be zero, i.e., $m_l(\xi) = 0$ and $m_g(\xi) = 0$, respectively.

- In the exploitation state denoted by $\xi(k) = 2$, the particles are supposed to exploit the region around personal best particles. To avoid premature convergence, randomly occurring distributed time-delays are added in the velocity updating

model, and a certain number of historical personal best particles are randomly selected for a more thorough search. In this case, the intensity factors are set as $m_l(\xi) = 0.01$ and $m_g(\xi) = 0$.

- In the exploration state denoted by $\xi(k) = 3$, the particles are encouraged to explore the entire search space thoroughly. Hence, randomly occurring distributed time-delays are added in the velocity updating model, and a certain number of historical global best particles are randomly selected with the intensity factors $m_l(\xi) = 0$ and $m_g(\xi) = 0.01$.

- In the jumping-out state denoted by $\xi(k) = 4$, the particles are trying to escape from the region around the local optimum. Therefore, distributed time-delays are added in the velocity updating model where a certain number of historical personal and global best particles are randomly selected with the intensity factors $m_l(\xi) = 0.01$ and $m_g(\xi) = 0.01$.

The discussion of the above strategy can be summarized in Table 5.1, where the intensity factors $m_l(\xi)$ and $m_g(\xi)$ are determined by the evolutionary states; and k represents the number of current iteration.

Table 5.1 Velocity Updating Strategy for Distributed Time-delayed Information

State	Mode	$m_l(\xi)$	$m_g(\xi)$
Convergence	$\xi(k) = 1$	0	0
Exploitation	$\xi(k) = 2$	0.01	0
Exploration	$\xi(k) = 3$	0	0.01
Jumping-out	$\xi(k) = 4$	0.01	0.01

5.3 Results and Discussions

5.3.1 Selection of Benchmark Functions

In this chapter, eight well-known benchmark functions are employed to evaluate the performance of the proposed RODDPSO algorithm. The benchmark functions are

shown by (5.5) to (5.12). It should be pointed out that detailed information of the benchmark functions is displayed in Table 5.2 including the function number, the function name, the dimension, the search space of each dimension, the threshold, and the minimum of the benchmark functions.

Table 5.2 Configuration of benchmark functions

Functions	Name	Dimension	Search space	Threshold	Minimum
$f_1(x)$	Sphere	50	$[-100, 100]$	0.01	0
$f_2(x)$	Rosenbrock	50	$[-30, 30]$	100	0
$f_3(x)$	Ackley	50	$[-32, 32]$	0.01	0
$f_4(x)$	Rastrigin	50	$[-5.12, 5.12]$	50	0
$f_5(x)$	Schwefel 2.22	50	$[-10, 10]$	0.01	0
$f_6(x)$	Schwefel 1.2	50	$[-100, 100]$	0.01	0
$f_7(x)$	Griewank	50	$[-600, 600]$	0.01	0
$f_8(x)$	Step	50	$[-100, 100]$	0	0

Note that all the benchmark functions are high-dimensional problems. The Sphere function $f_1(x)$ is unimodal and is used to explore the convergence rate of the optimization problem. The Rosenbrock function $f_2(x)$ is a non-convex function which is also known as the Rosenbrock's banana function. The Ackley function $f_3(x)$ and the Rastrigin function $f_4(x)$ are very difficult to optimize because of a large number of local minima. The Schwefel 2.22 function $f_5(x)$ and the Schwefel 1.2 function $f_6(x)$ are classical unimodal and multimodal functions, which are hard to find the optimum. The Griewank function $f_7(x)$ is a popular benchmark function which is widely used to test the convergence of optimization algorithms. The step function $f_8(x)$ is also a typical benchmark function. Here, $x = (x_1, x_2, \cdots, x_D)$ where D is the dimension of the search space. In our simulation, D is taken as 50.

$$\text{Sphere} : f_1(x) = \sum_{i=1}^{D} x_i^2. \tag{5.5}$$

$$\text{Rosenbrock} : f_2(x) = \sum_{i=1}^{D-1} (100(x_{i+1} - x_i)^2 + (x_i - 1)^2). \tag{5.6}$$

$$\begin{aligned}\text{Ackley} : f_3(x) = &-20e^{-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}} + 20 + e \\ &- e^{\frac{1}{D}\sum_{i=1}^{D} \cos 2\pi x_i}.\end{aligned} \tag{5.7}$$

$$\text{Rastrigin}: f_4(x) = \sum_{i=1}^{D}(x_i^2 - 10\cos 2\pi x_i + 10). \tag{5.8}$$

$$\text{Schwefel 2.22}: f_5(x) = \sum_{i=1}^{D} \mid x_i \mid + \prod_{i=1}^{D} \mid x_i \mid . \tag{5.9}$$

$$\text{Schwefel 1.2}: f_6(x) = \sum_{i=1}^{D}(\sum_{j=1}^{i} x_j)^2. \tag{5.10}$$

$$\text{Griewank}: f_7(x) = 1 + \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}}). \tag{5.11}$$

$$\text{Step}: f_8(x) = \sum_{i=1}^{D}(\lfloor x_i + 0.5 \rfloor)^2. \tag{5.12}$$

5.3.2 Experiment Results of the RODDPSO Algorithm

As discussed above, eight benchmark functions are employed to evaluate the performance of the introduced RODDPSO algorithm. The superiority of the proposed RODDPSO algorithm is demonstrated over six popular PSO algorithms including the PSO-LDIW [135], PSO-TVAC [129], PSO-CK [27], SPSO [149] SDPSO [188] and MDPSO [140]. The parameters of the experiments are given as follows: the dimension of the search space is $D = 50$, and the population of the swarm is $S = 20$. It should be noted that each experiment has been repeated 20 times independently to avoid random influence. The setting of the distributed time-delay τ is determined based on the simulation results. The performance of the RODDPSO algorithm in the 20-dimensional search space with different settings of the upper bound of the distributed time-delay N is shown in Table 5.5. It can be seen that the RODDPSO algorithm demonstrates competitive performance when $N = 100$.

The performance tests for the proposed RODDPSO algorithms are shown in Fig. 5.2 to Fig. 5.9. The vertical coordinate represents the logarithmic formation of the mean fitness value of all the tested PSO algorithms, and the horizontal coordinate denotes the number of iteration for Fig. 5.2 to Fig. 5.9. Additionally, detailed information of the optimization performance is listed in Table 5.4, where the mean, the minimum, and the standard deviation of the fitness value with respect to each benchmark function is presented to demonstrate the performance of various PSO algorithms as well as the successful convergence ratio.

Table 5.3 Performance evaluation of RODDPSO algorithm with different N

		N=25	N=50	N=75	N=100	N=125	N=150	N=175	N=200
$f_1(x)$	Minimum	7.33×10^{-179}	0.0000	0.0000	0.0000	0.0000	8.95×10^{-319}	2.26×10^{-305}	8.94×10^{-281}
	Mean	1.12×10^{-142}	1.48×10^{-323}	4.94×10^{-324}	4.94×10^{-324}	0.0000	1.69×10^{-300}	4.45×10^{-260}	3.52×10^{-236}
	Std. Dev.	4.80×10^{-142}	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	Ratio	100%	100%	100%	100%	100%	100%	100%	100%
$f_2(x)$	Minimum	2.03×10^{-2}	4.19×10^{-3}	2.28×10^{-2}	2.29×10^{-4}	1.16×10^{-3}	2.61×10^{-4}	1.60×10^{-5}	5.14×10^{-6}
	Mean	8.0533	5.1520	1.60×10^{2}	6.4467	1.02×10^{1}	1.45×10^{1}	1.03×10^{1}	7.5699
	Std. Dev.	4.5765	4.4331	6.75×10^{2}	6.2135	1.47×10^{1}	2.01×10^{1}	1.46×10^{1}	4.9529
	Ratio	100%	100%	95%	100%	100%	100%	100%	100%
$f_3(x)$	Minimum	2.66×10^{-15}	2.66×10^{-15}	2.66×10^{-15}	2.66×10^{-15}	2.66×10^{-15}	2.66×10^{-15}	2.66×10^{-15}	2.66×10^{-15}
	Mean	6.04×10^{-15}	5.15×10^{-15}	4.26×10^{-15}	4.80×10^{-15}	5.15×10^{-15}	4.26×10^{-15}	5.68×10^{-15}	4.80×10^{-15}
	Std. Dev.	7.94×10^{-16}	1.67×10^{-15}	1.81×10^{-15}	1.79×10^{-15}	1.67×10^{-15}	1.81×10^{-15}	1.30×10^{-15}	1.79×10^{-15}
	Ratio	100%	100%	100%	100%	100%	100%	100%	100%
$f_4(x)$	Minimum	4.9748	5.9698	6.9647	4.9748	4.9748	4.9748	4.9748	4.9748
	Mean	1.02×10^{1}	1.04×10^{1}	1.05×10^{1}	1.12×10^{1}	1.02×10^{1}	1.06×10^{1}	8.9049	9.8998
	Std. Dev.	2.8345	2.9316	3.0874	3.7350	3.2422	3.8096	2.3384	2.9844
	Ratio	100%	100%	100%	100%	100%	100%	100%	100%
$f_5(x)$	Minimum	5.40×10^{-48}	4.34×10^{-60}	9.26×10^{-76}	3.17×10^{-77}	4.56×10^{-86}	1.28×10^{-86}	2.22×10^{-102}	1.04×10^{-86}
	Mean	1.15×10^{-31}	1.83×10^{-33}	6.83×10^{-44}	1.63×10^{-58}	6.02×10^{-59}	4.28×10^{-52}	4.04×10^{-59}	8.72×10^{-56}
	Std. Dev.	3.43×10^{-31}	7.91×10^{-33}	2.10×10^{-43}	4.94×10^{-58}	2.07×10^{-58}	1.48×10^{-51}	1.75×10^{-58}	3.89×10^{-55}
	Ratio	100%	100%	100%	100%	100%	100%	100%	100%
$f_6(x)$	Minimum	1.47×10^{-32}	1.39×10^{-45}	1.84×10^{-60}	1.78×10^{-66}	2.50×10^{-67}	2.35×10^{-67}	1.18×10^{-62}	9.19×10^{-59}
	Mean	1.23×10^{-22}	2.28×10^{-34}	1.88×10^{-45}	2.37×10^{-55}	2.39×10^{-53}	5.43×10^{-53}	6.71×10^{-52}	1.64×10^{-47}
	Std. Dev.	2.57×10^{-22}	6.83×10^{-34}	8.35×10^{-45}	5.03×10^{-55}	1.07×10^{-52}	2.12×10^{-52}	3.00×10^{-51}	6.23×10^{-47}
	Ratio	100%	100%	100%	100%	100%	100%	100%	100%
$f_7(x)$	Minimum	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	Mean	3.77×10^{-2}	3.10×10^{-2}	4.47×10^{-2}	4.90×10^{-2}	3.32×10^{-2}	1.83×10^{-2}	1.96×10^{-2}	2.86×10^{-2}
	Std. Dev.	3.24×10^{-2}	2.38×10^{-2}	3.47×10^{-2}	4.34×10^{-2}	3.22×10^{-2}	1.96×10^{-2}	2.68×10^{-2}	2.11×10^{-2}
	Ratio	25%	15%	10%	15%	15%	45%	50%	20%
$f_8(x)$	Minimum	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	Mean	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	Std. Dev.	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	Ratio	100%	100%	100%	100%	100%	100%	100%	100%

It can be seen that the proposed RODDPSO algorithm demonstrates superiority over other PSO algorithms in terms of evaluation indices such as the mean, the minimum, and the standard deviation of the fitness values for function (5.5) to (5.12). Specifically, the mean fitness value of the RODDPSO algorithm is smaller than that of other PSO algorithms, which demonstrates the superiority of RODDPSO in reaching the global optimum. Moreover, although the RODDPSO algorithm cannot reach the best mean fitness for function (5.11), it presents competitive performance compared with the PSO-LDIW, PSO-TVAC, PSO-CK and SPSO algorithms. Similarly, the RODDPSO algorithm outperforms the PSO-LDIW, PSO-CK, SPSO, and SDPSO algorithms for function (5.12) as shown in Fig. 5.9.

In addition to the mean fitness, the successful convergence ratio is a very important index to justify the convergence performance of optimization algorithms. The successful convergence ratio is not always 100% because the testing algorithms cannot always reach the global optimum for all the benchmark functions as shown in Table 5.4. Note that the RODDPSO algorithm demonstrates competitive performance over other PSO algorithms for function (5.5) to function (5.10) and function (5.12). Note that the

Table 5.4 Comparisons of various PSO algorithms on eight optimization benchmark functions

		PSO-LDIW	PSO-TVAC	PSO-CK	SPSO	SDPSO	MDPSO	RODDPSO
$f_1(x)$	Minimum	1.83×10^{-201}	5.19×10^{-159}	0.0000	6.35×10^{-177}	8.37×10^{-18}	3.59×10^{-102}	0.0000
	Mean	5.00×10^{2}	4.76×10^{-97}	5.00×10^{2}	5.00×10^{2}	3.85×10^{-11}	3.59×10^{-67}	9.88×10^{-324}
	Std. Dev.	2.24×10^{3}	1.76×10^{-96}	2.24×10^{3}	2.24×10^{3}	7.35×10^{-11}	1.60×10^{-66}	0.0000
	Ratio	95%	100%	95%	95%	100%	100%	100%
$f_2(x)$	Minimum	5.38×10^{-3}	1.2375	6.81×10^{-9}	2.70×10^{-6}	9.41×10^{-1}	1.53×10^{-2}	2.43×10^{-2}
	Mean	1.37×10^{4}	1.51×10^{1}	9.07×10^{3}	1.37×10^{4}	1.91×10^{1}	1.16×10^{1}	6.6373
	Std. Dev.	3.29×10^{4}	1.74×10^{1}	2.77×10^{4}	3.29×10^{4}	1.78×10^{1}	1.39×10^{1}	4.7374
	Ratio	75%	100%	75%	75%	100%	100%	100%
$f_3(x)$	Minimum	2.66×10^{-15}	2.66×10^{-15}	6.22×10^{-15}	6.22×10^{-15}	2.93×10^{-8}	6.22×10^{-15}	2.66×10^{-15}
	Mean	7.68×10^{-1}	5.68×10^{-15}	2.2544	2.9002	1.57×10^{-6}	6.93×10^{-15}	4.80×10^{-15}
	Std. Dev.	3.4329	1.30×10^{-15}	3.2030	3.3136	1.91×10^{-6}	2.19×10^{-15}	1.79×10^{-15}
	Ratio	95%	100%	15%	10%	100%	100%	100%
$f_4(x)$	Minimum	4.9748	3.9798	1.89×10^{1}	3.08×10^{1}	2.9850	6.9647	3.9798
	Mean	1.29×10^{1}	9.8501	5.49×10^{1}	6.60×10^{1}	1.93×10^{1}	1.11×10^{1}	9.5516
	Std. Dev.	1.34×10^{1}	4.0435	2.28×10^{1}	2.12×10^{1}	1.13×10^{1}	3.6845	3.0692
	Ratio	95%	100%	50%	25%	100%	100%	100%
$f_5(x)$	Minimum	8.46×10^{-121}	2.67×10^{-32}	5.74×10^{-34}	2.08×10^{-66}	4.22×10^{-9}	1.02×10^{-44}	8.79×10^{-80}
	Mean	1.65×10^{1}	1.01×10^{-19}	6.0000	9.0000	5.00×10^{-1}	1.69×10^{-28}	1.61×10^{-52}
	Std. Dev.	1.18×10^{1}	4.50×10^{-19}	6.8056	1.21×10^{1}	2.2361	7.18×10^{-28}	7.18×10^{-52}
	Ratio	20%	100%	50%	45%	95%	100%	100%
$f_6(x)$	Minimum	5.69×10^{-26}	1.92×10^{-29}	8.30×10^{-103}	5.11×10^{-53}	2.12×10^{-1}	5.30×10^{-28}	6.62×10^{-70}
	Mean	1.00×10^{3}	4.40×10^{-15}	1.25×10^{3}	2.33×10^{3}	1.8968	1.31×10^{-18}	2.42×10^{-48}
	Std. Dev.	2.05×10^{3}	1.97×10^{-14}	2.22×10^{3}	3.88×10^{3}	1.4415	4.23×10^{-18}	1.08×10^{-47}
	Ratio	80%	100%	75%	65%	0%	100%	100%
$f_7(x)$	Minimum	9.86×10^{-3}	0.0000	0.0000	0.0000	2.72×10^{-13}	0.0000	0.0000
	Mean	4.83×10^{-2}	3.22×10^{-2}	5.29×10^{-2}	4.5869	1.88×10^{-2}	2.05×10^{-2}	2.80×10^{-2}
	Std. Dev.	2.79×10^{-2}	3.51×10^{-2}	1.19×10^{-1}	2.03×10^{1}	1.55×10^{-2}	2.43×10^{-2}	2.56×10^{-2}
	Ratio	5%	25%	40%	20%	30%	40%	30%
$f_8(x)$	Minimum	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	Mean	0.0000	0.0000	5.01×10^{2}	6.5500	0.0000	0.0000	0.0000
	Std. Dev.	0.0000	0.0000	2.24×10^{3}	2.70×10^{1}	0.0000	0.0000	0.0000
	Ratio	100%	100%	80%	85%	100%	100%	100%

Griewank function has a very large number of local minima, therefore, it is difficult to detect the global minimum, which leads to a low successful convergence ratio. We can see that all the testing PSO algorithms have low successful convergence ratio for function (5.11) which are 5%, 25%, 40%, 20%, 30%, 40% and 30%, respectively. Nevertheless, the proposed RODDPSO algorithm can still reach the global minimum with a satisfactory mean fitness value, which demonstrates its competitive performance than other PSO algorithms.

The plots of the convergence rate for testing algorithms are depicted in Fig. 5.2 to Fig. 5.9. It is clear that the convergence rate of the RODDPSO algorithm is not as fast as the PSO-TVAC algorithm and the SDPSO algorithm at the beginning for function (5.5), however, the RODDPSO algorithm reaches the global optimum with better mean fitness value than other PSO algorithms. Moreover, it can be seen that the

RODDPSO algorithm tends to reach the global optimum robustly for all the benchmark functions according to the low mean fitness value and high successful convergence ratio. The proposed RODDPSO algorithm outperforms six popular PSO algorithms in both unimodal and multimodal optimization benchmark functions, which indicate that the RODDPSO algorithm is capable of getting rid of local optima. As such, the RODDPSO algorithm can solve the optimization problem with satisfactory convergence speed and convergence accuracy.

The ranking method is applied to quantitatively evaluate the convergence performance of the RODDPSO algorithm, where the ranking results are displayed in Table 5.5. It can be seen that the mean ranking value of the RODDPSO algorithm is 1.5, which is the smallest one among all the selected PSO algorithms. To summarise, the convergence behavior of the RODDPSO algorithm outperforms other selected PSO algorithms.

Table 5.5 Convergence performance evaluation by using the ranking method

	PSO-LDIW	PSO-TVAC	PSO-CK	SPSO	SDPSO	MDPSO	RODDPSO
$f_1(x)$ Ranking	5	2	5	5	4	3	1
$f_2(x)$ Ranking	6	3	5	6	4	2	1
$f_3(x)$ Ranking	5	2	6	7	4	3	1
$f_4(x)$ Ranking	4	2	6	7	5	3	1
$f_5(x)$ Ranking	7	3	5	6	4	2	1
$f_6(x)$ Ranking	5	3	6	7	4	2	1
$f_7(x)$ Ranking	5	4	6	7	1	2	3
$f_8(x)$ Ranking	5	2	7	8	1	4	3
Average Ranking	5.25	2.625	5.75	6.375	3.375	2.625	1.5

To comprehensively evaluate the performance of the proposed RODDPSO algorithm, the 50-dimensional search space is utilized. In this case, a series of experiments have been conducted to evaluate the effectiveness of the proposed RODDPSO algorithm where the search space is $D = 50$, the upper bound of the distributed time-delay is $N = 110$, and other parameter settings remain the same. The corresponding experiment results are displayed in Table 5.6.

We can see that the successful convergence ratio of the RODDPSO algorithm is satisfactory. Moreover, the proposed RODDPSO algorithm demonstrates competitive performance over other PSO algorithms in terms of the mean, the minimum, and

Table 5.6 Comparisons of various PSO algorithms in 50-dimensional search space

		PSO-LDIW	PSO-TVAC	PSO-CK	SPSO	SDPSO	MDPSO	RODDPSO
$f_1(x)$	Minimum	4.03×10^{-51}	1.58×10^{-14}	1.55×10^{-69}	2.63×10^{-140}	1.51×10^{-1}	4.24×10^{-27}	2.35×10^{-118}
	Mean	3.00×10^{3}	7.29×10^{-10}	5.50×10^{3}	2.50×10^{3}	9.3336	2.63×10^{-19}	7.29×10^{-102}
	Std. Dev.	4.70×10^{3}	1.49×10^{-9}	7.59×10^{3}	4.44×10^{3}	9.8519	1.11×10^{-18}	2.87×10^{-101}
	Ratio	70%	100%	60%	75%	0%	100%	100%
$f_2(x)$	Minimum	5.0985	1.65×10^{1}	1.23×10^{-5}	4.6525	1.83×10^{2}	4.8925	4.66×10^{-5}
	Mean	4.76×10^{3}	1.40×10^{2}	4.00×10^{6}	4.91×10^{3}	1.02×10^{3}	5.95×10^{1}	1.91×10^{2}
	Std. Dev.	2.01×10^{4}	1.41×10^{2}	1.79×10^{7}	2.01×10^{4}	9.55×10^{2}	4.41×10^{1}	6.67×10^{2}
	Ratio	70%	50%	90%	60%	0%	85%	95%
$f_3(x)$	Minimum	1.33×10^{-14}	3.85×10^{-6}	4.1669	1.33×10^{-14}	7.44×10^{-1}	1.72×10^{-12}	2.04×10^{-14}
	Mean	5.2116	1.4500	1.11×10^{1}	4.6816	1.8167	1.32×10^{-6}	5.84×10^{-1}
	Std. Dev.	6.6150	2.6486	4.3265	4.8179	5.48×10^{-1}	5.74×10^{-6}	7.42×10^{-1}
	Ratio	60%	35%	0%	10%	0%	100%	60%
$f_4(x)$	Minimum	4.08×10^{1}	5.27×10^{1}	1.38×10^{2}	1.07×10^{2}	8.98×10^{1}	3.48×10^{1}	4.68×10^{1}
	Mean	1.21×10^{2}	8.86×10^{1}	2.23×10^{2}	1.76×10^{2}	1.38×10^{2}	7.62×10^{1}	7.45×10^{1}
	Std. Dev.	5.57×10^{1}	2.18×10^{1}	4.28×10^{1}	3.19×10^{1}	3.93×10^{1}	2.10×10^{1}	1.45×10^{1}
	Ratio	5%	0%	0%	0%	0%	10%	5%
$f_5(x)$	Minimum	2.00×10^{1}	2.34×10^{-6}	2.76×10^{-2}	1.97×10^{-60}	2.23×10^{-1}	1.28×10^{-12}	3.12×10^{-17}
	Mean	5.85×10^{1}	2.5012	2.86×10^{1}	3.70×10^{1}	2.49×10^{1}	1.0000	1.0000
	Std. Dev.	2.54×10^{1}	4.4421	1.85×10^{1}	1.56×10^{1}	2.14×10^{1}	3.0779	3.0779
	Ratio	0%	70%	0%	10%	0%	90%	90%
$f_6(x)$	Minimum	5.01×10^{3}	3.2281	6.59×10^{-11}	1.58×10^{-4}	2.84×10^{3}	3.00×10^{-1}	4.49×10^{-4}
	Mean	2.58×10^{4}	3.94×10^{3}	1.22×10^{4}	1.67×10^{4}	1.30×10^{4}	1.11×10^{3}	5.01×10^{2}
	Std. Dev.	1.41×10^{4}	3.80×10^{3}	1.01×10^{4}	1.92×10^{4}	7.15×10^{3}	2.30×10^{3}	1.54×10^{3}
	Ratio	0%	0%	15%	25%	0%	0%	30%
$f_7(x)$	Minimum	0.0000	1.76×10^{-12}	1.01×10^{-14}	0.0000	3.92×10^{-1}	0.0000	1.11×10^{-16}
	Mean	4.52×10^{1}	2.37×10^{-2}	4.54×10^{1}	9.1116	8.81×10^{-1}	3.30×10^{-2}	2.13×10^{-2}
	Std. Dev.	6.22×10^{1}	2.96×10^{-2}	6.88×10^{1}	2.78×10^{1}	2.28×10^{-1}	3.43×10^{-2}	2.80×10^{-2}
	Ratio	35%	50%	15%	40%	0%	35%	45%
$f_8(x)$	Minimum	0.0000	0.0000	1.30×10^{1}	0.0000	6.0000	0.0000	0.0000
	Mean	1.50×10^{3}	2.50×10^{-1}	7.38×10^{3}	2.01×10^{3}	1.41×10^{1}	0.0000	2.50×10^{-1}
	Std. Dev.	3.66×10^{3}	5.50×10^{-1}	6.59×10^{3}	4.10×10^{3}	6.2146	0.0000	4.44×10^{-1}
	Ratio	85%	80%	0%	15%	0%	100%	75%

the standard deviation of the fitness value via the selected benchmark functions. Therefore, the proposed RODDPSO algorithm exhibits satisfactory performance on the convergence, accuracy and the success ratio in the 50-dimensional search space.

5.4 Conclusion

In this chapter, a novel RODDPSO algorithm is proposed with hope to improve the search ability of the PSO algorithm. The velocity updating model of the RODDPSO algorithm is adaptively adjusted depending on the evolutionary state. It is worth mentioning that the distributed time-delay terms containing historical information of previous personal and global best particles are added in the velocity updating model. As such, the RODDPSO algorithm is capable of escaping from local optima, and the search space is explored and exploited more thoroughly than the classic PSO algorithm. The

superiority of the proposed RODDPSO algorithm is demonstrated over six well-known PSO algorithms on eight popular benchmark functions including both unimodal and multimodal cases. In the future, we aim to: 1) further improve the convergence speed of the RODDPSO algorithm; 2) extend our results to multi-objective optimization [58, 174, 69, 184, 99]; and 3) apply our RODDPSO algorithm to practical applications, such as telecommunication, power systems, healthcare, and control systems.

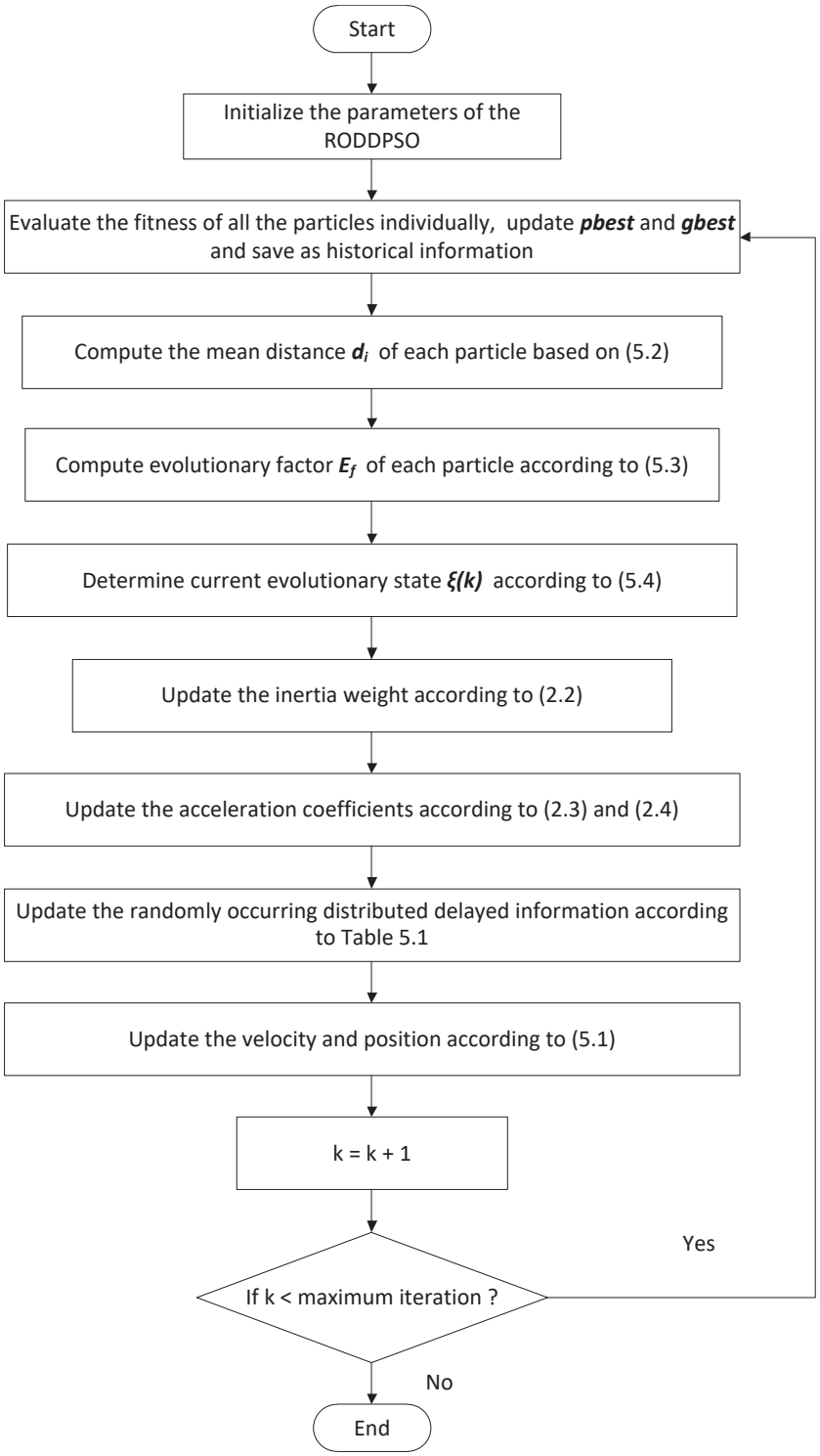

Fig. 5.1 The flowchart of the RODDPSO algorithm

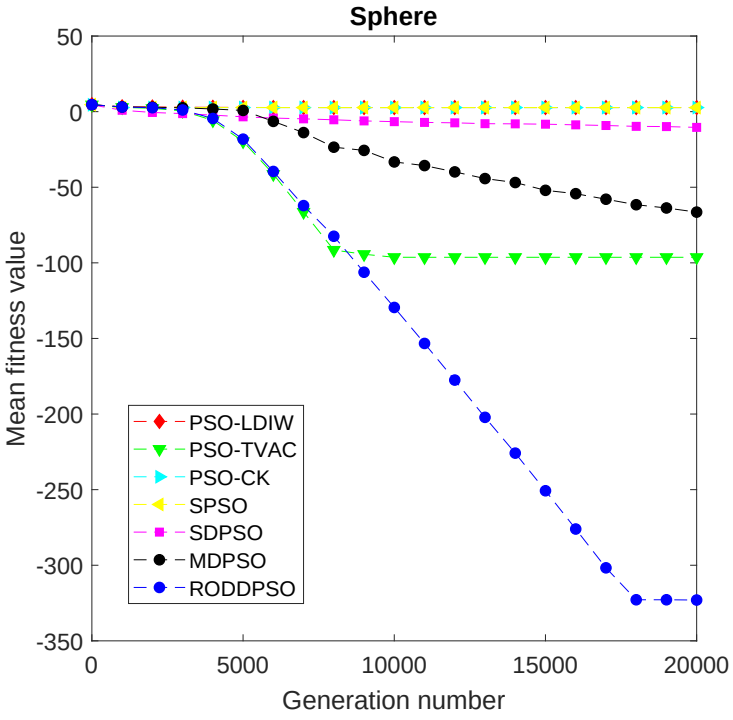

Fig. 5.2 Performance test for Sphere function $f_1(x)$

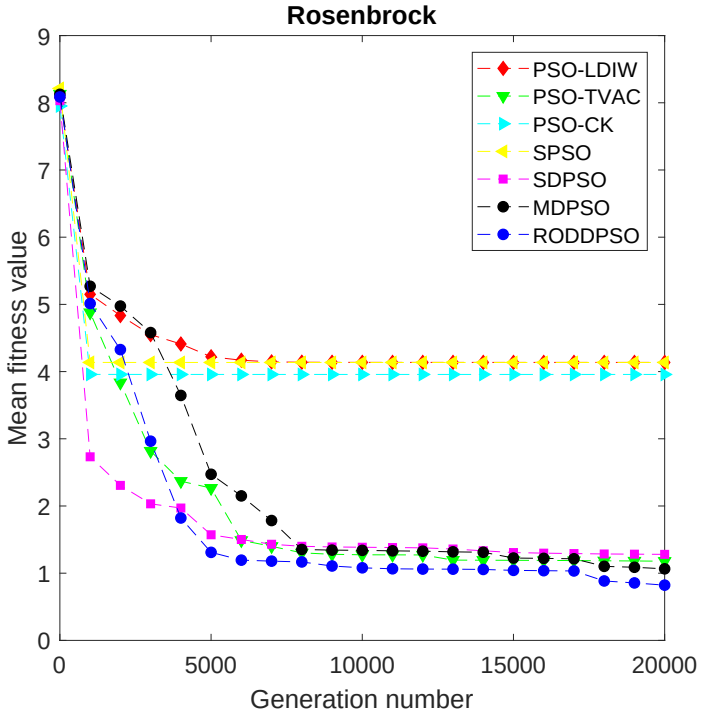

Fig. 5.3 Performance test for Rosenbrock function $f_2(x)$

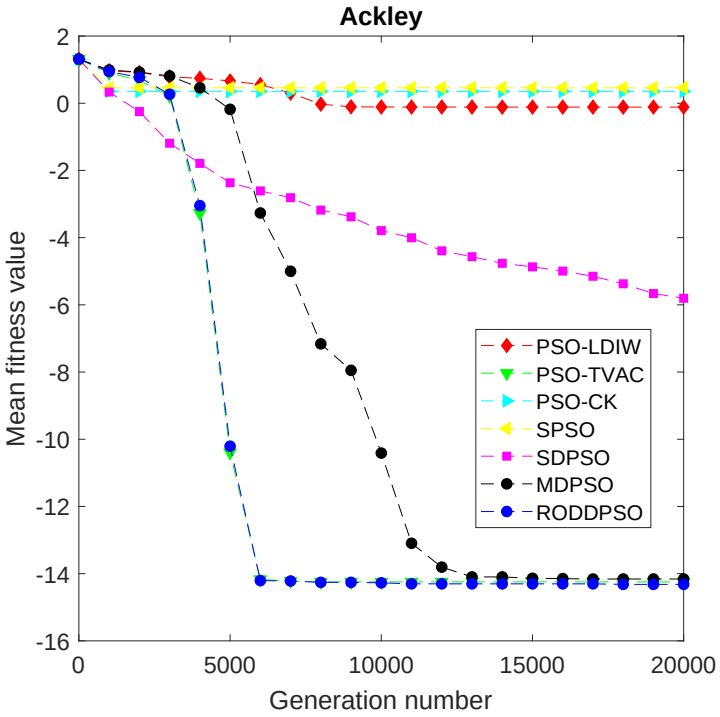

Fig. 5.4 Performance test for Ackley function $f_3(x)$

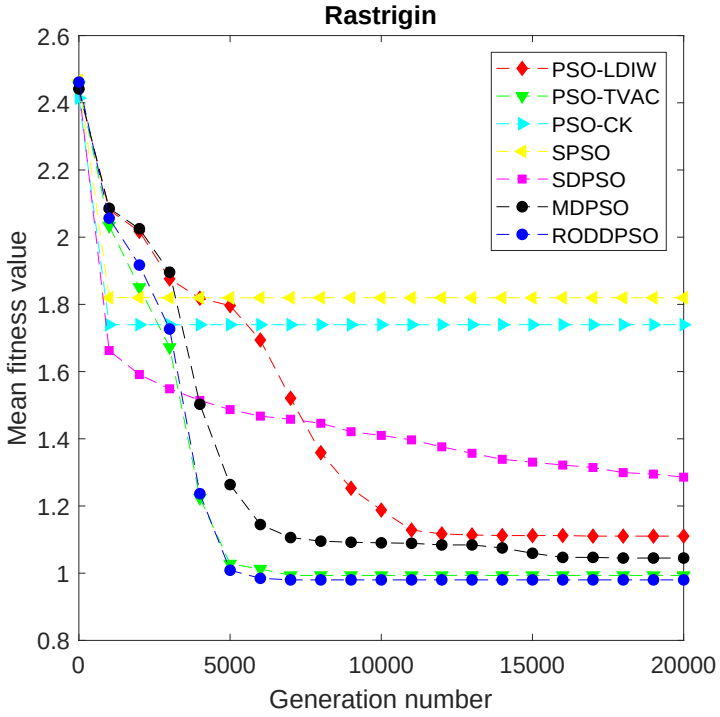

Fig. 5.5 Performance test for Rastrigin function $f_4(x)$

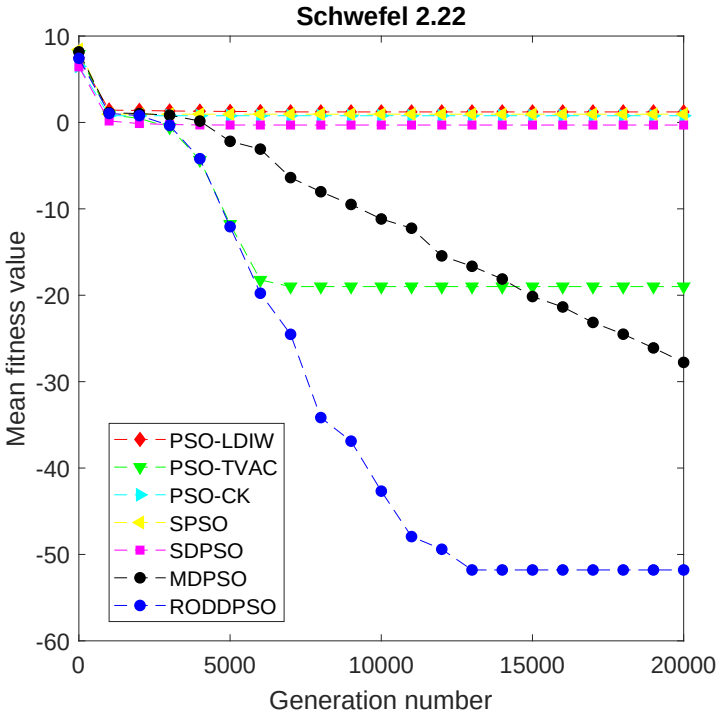

Fig. 5.6 Performance test for Schwefel 2.22 function $f_5(x)$

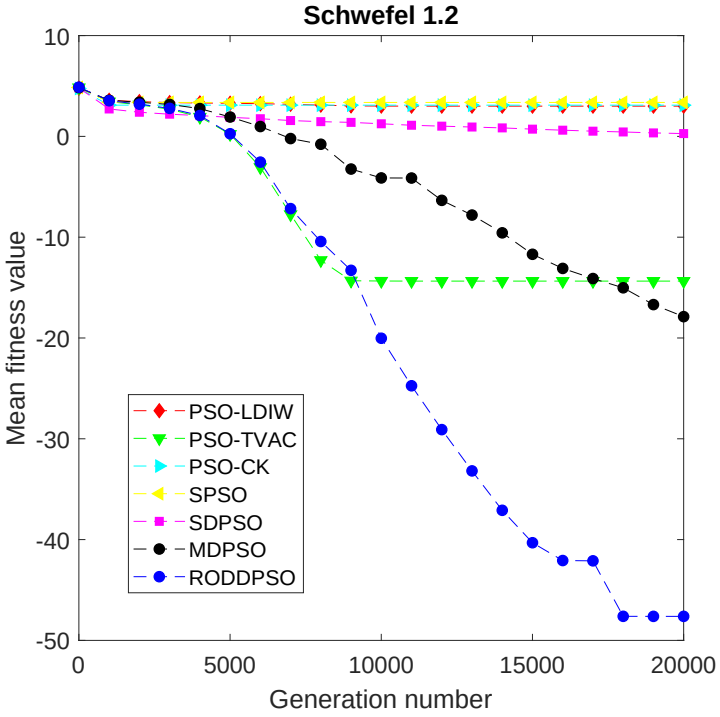

Fig. 5.7 Performance test for Schwefel 1.2 function $f_6(x)$

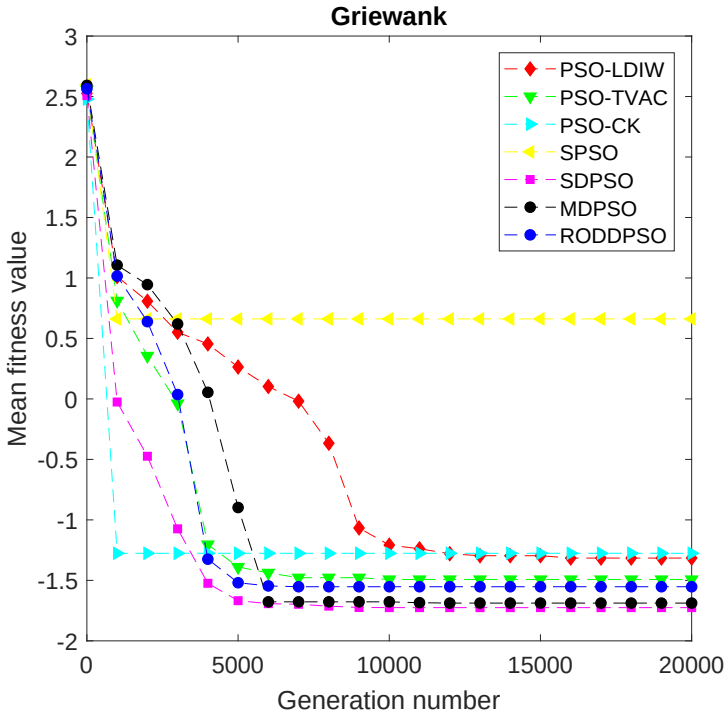

Fig. 5.8 Performance test for Griewank function $f_7(x)$

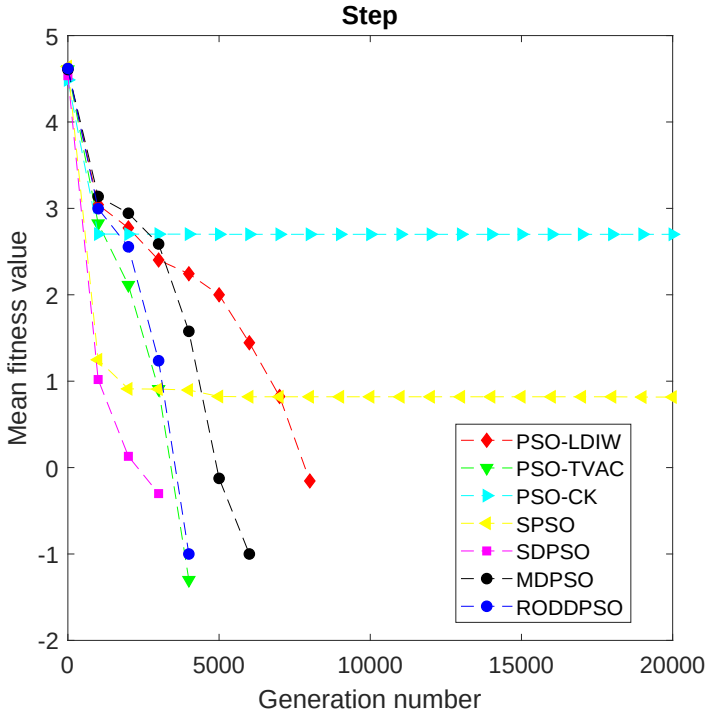

Fig. 5.9 Performance test for Step function $f_8(x)$

Chapter 6

Applications of Particle Swarm Optimization Algorithms in Intelligent Data Analysis

6.1 Motivation

Accident & emergency (A&E) departments in National Health Service (NHS) in the UK are open for 24 hours and 365 days a year. Targets for A&E departments aim to ensure that at least 98% of patients are treated from arrival to discharge, transfer or admission within 4 hours. An obvious challenge is that patients requiring urgent treatment can go straight to the A&E at any time, thereby causing substantial strain on limited medical resources. Moreover, increasing numbers of emergency cases are leading to overcrowding in many A&E departments. Consequently, A&E departments suffer from financial pressures [3, 155]. Furthermore, a number of non-emergency patients go to the A&E departments, which leads to the increasing burden on the human and medical resources. Non-emergency patient groups could be targeted with other (non-emergency) services.

In A&E departments, an obvious challenge is that patients requiring urgent treatment can go straight to the A&E at any time, thereby causing substantial strain on limited medical resources. Hence, the number of emergency cases increases rapidly in

recent years, which leads to overcrowding in many A&E departments. In response to the revolution of data mining and machine learning techniques, it becomes more and more convenient for A&E staff to manage medical resources and arrange work schedules, thereby meeting the 4-hour requirement in emergency departments [15]. For instance, computer simulation models have been widely used for simulating real-world systems. Mathematical models have been introduced in [45, 28] to simulate the patient flow of emergency departments. A discrete-event simulation model has been introduced in [85] to simulate the patient flows in A&E departments, and multi-objective optimization analysis has been conducted for bed management. The relationship between ambient air pollution and patients' attendance at emergency departments has been studied in [68].

Moreover, overcrowding in A&E departments brings many adverse effects such as lower treatment quality, increased working burden and increased patient waiting time. Note that the identification of illness severity plays an important role in medical resources management. Grouping patients with an appropriate triage category is an important element in improving the efficiency of medical treatment. Therefore, it is of vital importance to identify the triage category of the patients, which can be treated as a clustering problem. Importantly, an appropriate triage category enables patients with serious illness or injury to be treated. Non-emergency cases can also be re-routed to other services in the health system. In addition, the management of medical resources can be allocated in an appropriate manner to reduce the financial cost. As such, the generation of an accurate triage category is important for A&E departments.

It is worth mentioning that the unplanned and non-urgent patients may cause the overcrowding problem which leads to long waiting time. In this situation, patients with severe illness may not be treated on time thus leading to negative patient outcomes [10, 16, 15]. It is of practical significance to reduce the number of unplanned and non-urgent patients and their length of stay at the A&E departments to save the financial costs and deliver high-quality service to the patients with severe illness [169, 117, 80]. Therefore, an efficient and accurate identification of the patient attendance disposal based on the patient diagnosis record is required to reduce inpatient services, which can be treated as a classification problem.

Recently, a large number of machine learning (ML) algorithms have been successfully employed in the healthcare informatics [41, 38, 130, 100, 4, 80, 146]. For example, the random forests and the gradient boosted decision tree algorithm have been utilized in [130] in order to predict clinical outcomes (critical care and hospitalization) in A&E departments. In [100], an ensemble learning-based scoring system has been developed to predict acute cardiac complications for patients with chest pain in A&E departments. Very recently, the popular deep learning techniques (also known as deep neural networks (DNNs)) have been recognized as competitive ML approaches due to their promising performance in dealing with complex data with high-dimensionality, and the deep learning techniques have been successfully used in a wide range of research fields, e.g., signal processing, telecommunication, healthcare informatics, natural language processing and computer vision [65, 204, 55, 87].

To summarize, we focus on the clustering and classification problems in A&E data which are shown as follows: 1) the patient clustering problem for generating an accurate triage category for patients attending the A&E departments; and 2) the patient classification problem for efficiently and accurately identifying the patient attendance disposal for patients in A&E departments. For the first problem, a novel RODDPSO-based clustering algorithm is put forward which combines the proposed RODDPSO algorithm with the traditional K-means clustering algorithm, and successfully employed to analyze the A&E data in order to verify the triage categorization. For the second problem, a RODDPSO-based DBN is designed where the hyperparameters of the DBN is optimized by using the RODDPSO algorithm. The developed model is applied to the A&E data for classifying the patient attendance disposal categories.

The remainder of this chapter is organized as follows. The introduction of the A&E data is provided in Section 6.2. In Section 6.3, a novel RODDPSO-based clustering algorithm is proposed with hope to analyze the patient clustering problem in order to generate an appropriate triage category for all the patients attending the A&E department. In 6.4, the RODDPSO-based DBN is discussed and employed to classify the categories of the patient attendance disposal in A&E department. Finally, the conclusions of this chapter are drawn in Section 6.5.

6.2 A&E Data

6.2.1 Data Description

The data used in this thesis is provided by a hospital in West London including the urgent care center, the minor injury unit and the A&E department. The overall number of patient attendances at the emergency departments is 126986 over the period examined. Patient attendances at the A&E department, the urgent care center and the minor injury unit are 51713, 15151 and 60122, respectively. The detailed information of the three departments and the patient attendance are displayed in Table 6.1.

Table 6.1 Patients Attendance at the Emergency Departments

Department	Number of attendance
Accident and Emergency department	51713
Minor Injury Unit	15151
Urgent Care Centre	60122

In the raw data, each record represents an incident in a single row and each column indicates an attribute with respect to the patient. Note that there are totally 25 attributes in the data consisting of the pseudo NHS number, general practitioner (GP) practice code, patient age, arrival time, departure time, provider code, provider name, date time for treatment, fiscal year label, arrival month, arrival date, modal of arrival, mode of arrival description, attendance disposal, attendance disposal description, core healthcare resource group (HRG), HRG description, referral source, referral source description, A&E department description, clinical commissioning groups (CCGs), first diagnosis, diagnosis description, and postcode sector of usual address. The data is recorded in real-time, especially the arrival date time, conclusion date time and date time seen for treatment. Hence, we compute the time interval of treatment time and waiting time in A&E departments for later analysis.

6.3 Patient Clustering in A&E Data

On one research front with data analysis, clustering techniques have been successfully employed in a variety of research areas such as biology, signal processing, computer vision, market segmentation, and healthcare, see e.g., [146, 141, 52, 152, 39]. Clustering techniques are used to discover the natural groupings of a set of objects where the objects in the same cluster share similar characteristics. It has been shown in [32, 119] that many popular clustering algorithms are heavily dependent on the initial state of cluster centroids, and may get trapped in local optima. As such, it is reasonable to *optimize* the parameters of clustering algorithms (e.g. the number of clusters and the initial state of cluster centroids) in order to improve the clustering performance. In this context, various optimization algorithms have been applied to optimally set the parameters with examples including the genetic algorithm [84, 63], the simulated annealing algorithm [133, 128], the PSO algorithm [153, 78], and the artificial bee colony [195] algorithm.

PSO algorithms have proven to be a strong competitor to other optimization algorithms [6, 153, 173, 77, 34, 105]. For instance, a PSO-based clustering technique has been proposed in [153] where the initial swarm adopts clusters formed by the K-means clustering algorithm. A hybrid PSO-based clustering algorithm has been developed in [173] for gene clustering by employing the self-organizing map algorithm. Recently, a hybrid fuzzy clustering algorithm on the basis of the conventional PSO algorithm and fuzzy C-means clustering algorithm has been proposed in [77] with satisfactory performance on several well-known benchmark data sets. Very recently, a density-based PSO algorithm has been introduced in [6] for data clustering by combining the kernel density estimation method with the PSO algorithm. In response to the aforementioned successful applications that use PSO to improve clustering algorithms, a seemingly natural idea is to introduce the proposed RODDPSO approach to the clustering problem, which is then tested on the patient clustering problem using A&E hospital attendance data.

In this section, a novel RODDPSO-based clustering algorithm is devised by employing the proposed RODDPSO algorithm to improve the basic K-means clustering algorithm. It is well known that the K-means clustering algorithm is a popular

clustering algorithm due to its low computation cost and simple implementation. In our work, the RODDPSO algorithm is used to optimize the cluster centroids where each particle consists of N_c cluster centroids in a single vector. Moreover, the proposed RODDPSO-based clustering algorithm is applied to evaluate the patients' triage category using A&E attendance data. Note that the clustering performance of the introduced RODDPSO-based clustering algorithm is evaluated by adopting the silhouette clustering validation method [131]. The triage category is defined to include 5 groups in [112] which are immediate resuscitation, very urgent, urgent, standard and non-urgent. Therefore, the number of clusters is five, and the clustering performance is evaluated by comparing the silhouette coefficients obtained by the K-means clustering algorithm and the fuzzy C-means (FCM) clustering algorithm with the RODDPSO-based clustering algorithms.

6.3.1 A RODDPSO-Based Clustering Algorithm

Objective Function

In our work, the purpose of the objective function is to minimize the average distance between the data points to their own centroids, and the objective function is given as follows:

$$J = \frac{\sum_{j=1}^{N_c} \left[\sum_{\forall P_t \in C_{ij}} dist(P_t, M_{ij})/N_p \right]}{N_c} \tag{6.1}$$

where N_c represents the number of clusters; C_{ij} denotes the jth cluster of the ith particle; M_{ij} represents the jth cluster centroid of the ith particle; P_t denotes the tth data point; $dist(P_t, M_{ij})$ represents the Euclidean distance between the data point P_t and its cluster centroid M_{ij}; N_p represents the number of data points belonging to cluster C_{ij}; and N_c denotes the number of clusters.

Framework of the RODDPSO-Based Clustering Algorithm

The RODDPSO algorithm is used to optimize the cluster centroids in order to improve the clustering performance. It is worth mentioning that the powerful search ability of the proposed RODDPSO can reduce the possibility of being trapped in local

optima, and hence improve the clustering performance. The procedure of the proposed RODDPSO-based clustering algorithm is demonstrated in Algorithm 1.

Algorithm 1 RODDPSO-Based Clustering Algorithm

1. Initialize the parameters including the population size P, the velocity and position of the particles v_i, x_i, acceleration coefficients c_1, c_2, inertia weight w, maximum iteration, the number of clusters N_c, maximum velocity $V_{\max}$ and intensity factors m_i, m_g.
2. Randomly initialize every particle to contain N_c cluster centroids.
3. Calculate the Euclidean distance $dist(P_t, M_{ij})$ between the data point and its cluster centroid.
4. Assign the data points to the closest cluster.
5. Calculate the fitness of all particles based on the objective function (6.1).
6. Select the personal best particle and the global best particle.
7. Confirm the evolutionary state depending on the calculated evolutionary factor.
8. Update the velocity and position equations based on the evolutionary state according to (5.1).
9. Repeat Steps 3 to 8 till the algorithm reaches the maximal number of iterations.

6.3.2 Results and Discussions

Data Pre-processing

It should be noticed that the data includes missing values and redundant information. Hence, 3,778 incidents are removed from the dataset because their treatment date time is null or missing. Furthermore, redundant information is also removed, e.g., healthcare resource group (HRG) and HRG description, where the latter only represents the description of previous attribute. In addition, the irrelevant attributes such as the provider code and the GP practice code are also abandoned by employing statistical analysis. In this part, 10 attributes are selected as the inputs of the clustering algorithms: year, month, week, day, age, mode of arrival, first diagnosis, emergency attendance disposal, treatment time, and HRG. Furthermore, to avoid the influence of the attributes with larger number of values, the data is normalized according to the following equation:

$$X_{\text{Norm}} = \frac{x - \min(X)}{\max(X) - \min(X)} \tag{6.2}$$

where X_{Norm} represents the normalized value, $\min(X)$ represents the minimum value of data X, $\max(X)$ describes the maximum value of data X.

Cluster Evaluation

In data mining, the cluster evaluation is of vital importance in the cluster analysis, and thus we can

1) determine if there exist any non-random structures in the data;

2) determine the appropriate number of clusters;

3) evaluate the performance of clustering algorithms without reference to external information; and

4) determine the clustering performance by comparing the obtained results to some externally known results such as the class labels provided by experts.

In general, there exist three types of evaluation methods to judge various aspects of cluster validity: 1) unsupervised cluster evaluation, 2) supervised cluster evaluation, 3) and relative cluster evaluation. In our work, the unsupervised cluster evaluation approach is adopted to evaluate the clustering performance. The measures that the unsupervised cluster evaluation approach uses include cohesion and separation. The cohesion measure determines how the data points in a cluster are closely related and the separation measure reveals how a cluster is distinct from others.

Silhouette is a popular cluster evaluation method proposed in [131]. The value of the silhouette coefficient varies between -1 and 1. The positive values show that the data points are very far from neighboring clusters, and negative values represent that the data points are probably assigned to the wrong cluster. If the silhouette coefficient is 0, it means the data point lies equally far away from two clusters and can be assigned to any one of them. By taking the average of all the silhouette coefficients of data points, the average silhouette coefficients can be obtained. Generally, the overall clustering performance is evaluated by the average silhouette coefficients.

Experiment Results

In our work, we assume that the distribution of the collected data obeys the Gaussian distribution. To evaluate the clustering performance of the proposed RODDPSO-based clustering algorithm, we compare the silhouette coefficients of the RODDPSO-based clustering algorithm with the K-means and FCM clustering algorithms. In this part, the squared Euclidean (sqeuclidean) distance metric is adopted due to its simple implementation. The MATLAB plots of the silhouette coefficients of the K-means, the FCM and the RODDPSO-based clustering algorithms are depicted in Fig. 6.1, Fig. 6.2 and Fig. 6.3, respectively.

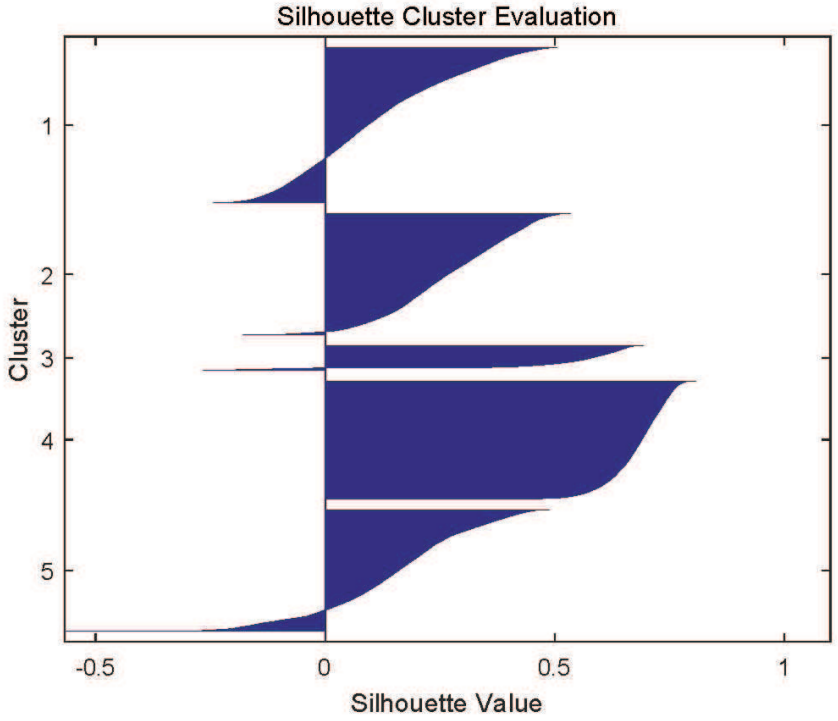

Fig. 6.1 Silhouette coefficient of K-means clustering algorithm

The mean silhouette coefficients of the K-means, the FCM and the RODDPSO-based clustering algorithms are 0.2970, 0.1253 and 0.3166, respectively. We can see that in Fig. 6.1, most of the silhouette values of the K-means clustering algorithm are positive, which indicates that most of the data points are assigned to the proper clusters. In Fig. 6.2, more than half of the data points obtain negative values of the silhouette coefficients, and the mean silhouette coefficient is much smaller than

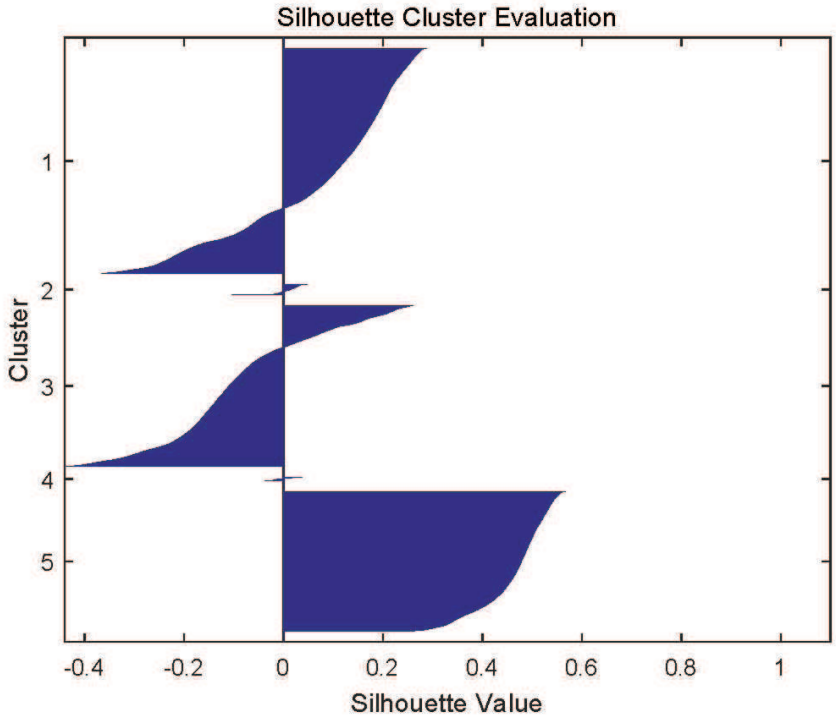

Fig. 6.2 Silhouette coefficient of Fuzzy C-means clustering algorithm

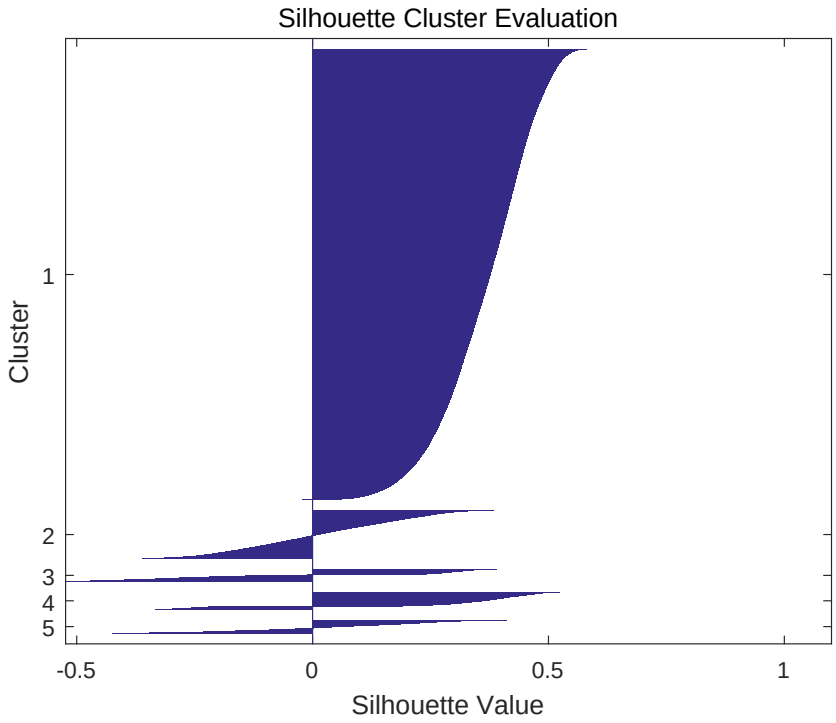

Fig. 6.3 Silhouette coefficient of RODDPSO-based clustering algorithm

that of the K-means and the RODDPSO-based clustering algorithms. As such, the clustering performance of FCM algorithm is not satisfactory. It has been shown in Fig. 6.3 that the mean silhouette value of the RODDPSO-based clustering algorithm is 0.3166 which is higher than the results of the K-means and the FCM clustering techniques. Furthermore, it is clear that there are fewer negative silhouette values using the RODDPSO-based clustering algorithm than the K-means and the FCM clustering techniques, which indicate fewer data points are assigned to the inappropriate clusters. Thus, the superiority and feasibility of the proposed RODDPSO-based clustering algorithm is demonstrated and the generated triage category is reasonable.

6.4 Patient Classification in A&E Data

In this section, the RODDPSO algorithm proposed in Chapter 5 is employed to optimize the hyperparameters of the DBN. With the optimized hyperparameters, the DBN is effectively and efficiently trained which results in better classification accuracy. The proposed RODDPSO-based DBN is successfully applied to analyze the A&E data so as to investigate the patient attendance disposal problem. The patient routing, the efficiency of both human and non-human resource management in A&E departments can be improved with a proper identification of the patient discharge, transfer or admission conditions.

6.4.1 Deep Belief Network and Restricted Boltzmann Machine

It is well known that the DBN proposed in 2006 has been recognized as the breakthrough of the deep learning [73, 65, 64, 11, 123, 55]. The DBN is stacked by a series of simple learning modules which are the restricted Boltzmann machines (RBMs) where each RBM is composed of a visible layer and a hidden layer. In this section, we will discuss about the main concepts of the DBN and RBM.

Restricted Boltzmann Machine

The RBMs are undirected probabilistic graphical models which are composed to build DBNs. RBMs are capable of learning the probability distribution with respect to the input data [115, 37]. An RBM is composed of one hidden layer and one visible layer. The hidden layer represents the features, and the visible layer represent the input data. A hidden layer consists of a number of binary stochastic hidden units represented by a hidden vector h, and a visible layer consists of several binary stochastic visible units represented by a visible vector v. In one visible layer, all the visible units are fully connected to all the hidden units in the hidden layer with certain weights. It should be mentioned that there are no inner connections among the units in the same layer. The schematic diagram of an RBM is depicted in Fig. 6.4.

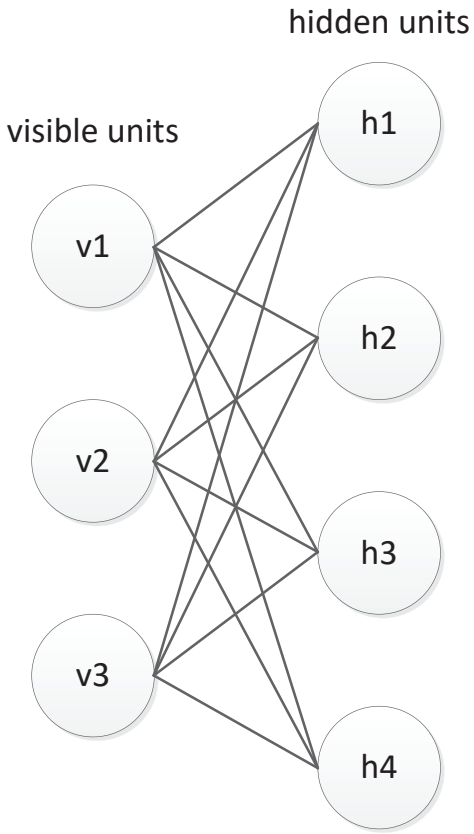

Fig. 6.4 The architecture of an RBM

An energy-based model, the energy function of an RBM is defined by the following function:

$$E(v, h|\theta) = -\sum_{i=1}^{m}\sum_{j=1}^{n} w_{ij} v_i h_j - \sum_{i=1}^{m} b_i v_i - \sum_{j=1}^{n} c_j h_j \tag{6.3}$$

where $\theta = (w, b, c)$ represent the parameters of an RBM. In this RBM, the weight (also known as the symmetric interaction term) between the visible unit v_i and the hidden unit h_j is defined by w_{ij}. The number of hidden and visible units are represented by n and m, respectively. The bias of the visible unit v_i is denoted by b_i, and the bias of the hidden unit h_j is represented by c_j. The joint probability distribution $p(v, h|\theta)$ over the visible layer and the hidden layer is computed by the following function:

$$p(v, h|\theta) = \frac{e^{-E(v,h|\theta)}}{Z(\theta)} \tag{6.4}$$

where $Z(\theta)$ is the partition function which is also known as a normalizing constant that is given by summing over all possible configurations of the visible and hidden vectors. The partition function $Z(\theta)$ is defined by:

$$Z(\theta) = \sum_{v,h} e^{-E(v,h|\theta)} \tag{6.5}$$

The marginal probability of a visible vector v is defined as follows:

$$p(v|\theta) = \frac{\sum_{h} e^{-E(v,h|\theta)}}{Z(\theta)} \tag{6.6}$$

It is worth mentioning that there are no intra-layer connections in an RBM, and all the hidden and visible units are conditionally independent. Taking above discussions into consideration, the conditional probabilities $p(v|h, \theta)$ and $p(h|v, \theta)$ can be derived from the joint distribution by:

$$\begin{aligned} p(v_i = 1|h, \theta) &= \sigma(\sum_{j=1}^{n} w_{ij} h_j + b_i) \\ p(h_j = 1|v, \theta) &= \sigma(\sum_{i=1}^{m} w_{ij} v_i + c_j) \end{aligned} \tag{6.7}$$

where $\sigma(\cdot)$ is a sigmoid function shown as follows:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{6.8}$$

A fast learning algorithm named by the contrastive divergence (CD) algorithm has been proposed in [73] to train the RBMs. According to the CD algorithm, the parameters of an RBM are updated by the following functions:

$$\begin{aligned} \Delta w_{ij} &= \epsilon_1(\langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{rec}}) \\ \Delta b_i &= \epsilon_1(\langle v_i \rangle_{\text{data}} - \langle v_i \rangle_{\text{rec}}) \\ \Delta c_j &= \epsilon_1(\langle h_j \rangle_{\text{data}} - \langle h_j \rangle_{\text{rec}}) \end{aligned} \tag{6.9}$$

where ϵ_1 denotes the learning rate, $\langle \cdot \rangle_{\text{data}}$ represents the expectation with respect to the distribution of the input data, and $\langle \cdot \rangle_{\text{rec}}$ is the expectation with respect to the distribution defined by the reconstruction model.

Deep Belief Network

DBNs have drawn tremendous attention in the past few years. As mentioned previously, the DBNs are formed by stacking a series of RBMs and the schematic diagram of a DBN is shown in Fig. 6.5. The upper layers of a DBN are expected to extract high-level features, which explain the input data, and the lower layers extract low-level features from the input data. The learning algorithm of the DBNs is a greedy layer-wise unsupervised learning algorithm proposed by Hinton in [73].

The learning algorithm of the DBNs can be summarized into two stages: 1) the unsupervised pre-training stage; and 2) the supervised fine-tuning stage. In general, the pre-training process of a DBN is used to initialize the weights of a deep neural network. The RBMs are trained in the bottom-up manner layer by layer where the input of the upper RBM is provided by the output of the lower RBM. At the fine-tuning stage, an additional output layer is added to the deep neural network in order to predict the desired labels, and the parameters of network are further tuned by employing the back-propagation algorithm [115, 65, 11, 72].

The penalty term which is also called the weight decay term is added in the DBN with hope to penalize the network and prevent overfitting [65, 81, 156]. In this case, the capacity of the DBN is controlled by adding the weight decay term on the weights, and the performance of the model on the testing dataset can be improved. It is also well-known that the momentum is an important factor in training a deep neural network which is used to control the learning speed and smooth out the update of the weights during the training process [81]. The momentum and the penalty term are employed in both of the pre-training stage and the fine-tuning stage in order to control possible oscillations during the training process.

The model parameters $\theta = (w, b, c)$ in the pre-training process at iteration t are thus updated by the following equations:

$$\begin{aligned} \Delta w_{ij}^{t} &= m_1 \times \Delta w_{ij}^{t-1} - \lambda_1 \times w_{ij}^{t} \\ &\quad + \epsilon_1 \times (\langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{rec}}) \\ \Delta b_i^{t} &= m_1 \times \Delta b_i^{t-1} + \epsilon_1 \times (\langle v_i \rangle_{\text{data}} - \langle v_i \rangle_{\text{rec}}) \\ \Delta c_j^{t} &= m_1 \times \Delta c_j^{t-1} + \epsilon_1 \times (\langle h_j \rangle_{\text{data}} - \langle h_j \rangle_{\text{rec}}) \end{aligned} \tag{6.10}$$

where m_1 represents the momentum parameter in the pre-training process. ϵ_1 is the learning rate in the pre-training process. λ_1 denotes the weight decay parameter (which is used to penalize weights with large magnitude) in the pre-training process. After the pre-training process, the DBN is further fine-tuned by using the back-propagation algorithm with the purpose of improving the classification performance of the DBN. Additionally, the momentum is also used to make the learning process stable, and the penalty term is added on the weights to avoid overfitting at the fine-tuning stage.

6.4.2 A Novel RODDPSO-Based DBN

In this part, we utilize a recently proposed PSO algorithm, the RODDPSO algorithm, to select suitable hyperparameters for DBNs. As mentioned previously, the momentum and weight decay terms are employed in both of the pre-training and fine-tuning stages. In this case, there are six parameters we need to optimize in our work which are the momentum parameter for pre-training m_1, the momentum parameter for fine-tuning

m_2, the weight decay for pre-training λ_1, the weight decay for fine-tuning λ_2, the learning rate for pre-training ϵ_1, and the learning rate for fine-tuning ϵ_2. Therefore, the dimension of the search space is $D = 6$ and a candidate solution of our optimization problem is represented by a vector $\begin{bmatrix} m_1 & m_2 & \lambda_1 & \lambda_2 & \epsilon_1 & \epsilon_2 \end{bmatrix}$.

Objective Function

In our work, the developed approach aims to select a set of parameters to minimize the objective function of the RODDPSO algorithm, which is given as follows:

$$\hat{J} = \frac{1}{2}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 + \frac{\lambda_2}{2}\sum_{l=1}^{M-1}\sum_{j=1}^{m_{l+1}}\sum_{i=1}^{m_l} w_{ijl}^2 + \frac{\lambda_2}{2}\sum_{l=2}^{M}\sum_{i=1}^{m_l} d_{il}^2 \tag{6.11}$$

where y_i and $\hat{y}_i$ indicate the real class and the predicted class of the ith data point, respectively. N is the number of total data points. λ_2 represents the weight decay parameter for the fine-tuning process. M is the total number of layers in the DBN. m_l represents the number of neurons in the lth layer. w_{ijl} denotes the weight between the ith neuron in the lth layer and the jth neuron in the $(l+1)$th layer. d_{il} represents the threshold of the ith neurons in the lth layer. It should be mentioned that the objective function of the RODDPSO algorithm is also used as the loss function of the DBN at the fine-tuning stage.

Framework of the RODDPSO-Based DBN

It is worth mentioning that the RODDPSO algorithm is utilized to optimize the hyperparameters of a DBN in order to improve the classification performance. In our work, each particle (candidate solution) is represented by a vector which contains six hyperparameters. The training procedure of the introduced RODDPSO-based DBN is demonstrated in Algorithm 2.

Algorithm 2 The Training Algorithm of the RODDPSO-Based DBN

1. Initialize the parameters of the DBN and the RODDPSO algorithms (including the population size P, the velocity and position of the particles v_i, x_i, acceleration coefficients c_1, c_2, inertia weight w, maximum iteration number, the maximum value of velocity $V_{\max}$ and intensity factors m_i, m_g.)
2. Randomly initialize every particle containing the momentum terms m_1, m_2, the weight decay λ_1, λ_2, and the learning rates ϵ_1, ϵ_2.
3. Train the DBN and adjust the weighting parameters of the network based on (6.10).
4. Calculate the fitness of all particles according to the objective function (6.11).
5. Find the personal best particle and the global best particle.
6. Determine the evolutionary state according to the computed evolutionary factor.
7. Update the velocity and position equations according to (5.1).
8. Repeat Steps 3 to 7 till the algorithm reaches the maximal number of iterations.
9. Calculate the classification accuracy of the RODDPSO-based DBN with parameters in the global best particle.

6.4.3 Experiment Results and Discussions

Data Pre-processing

In this part, a well-known deep learning approach, the DBN is employed to investigate the classification problem on the patient attendance disposal. To comprehensively evaluate the performance of the proposed RODDPSO-based DBN, the standard DBN, the penalized DBN (with momentum and weight decay terms), and the KNN algorithms are used for comparison. It should be mentioned that the penalized DBN utilizes the momentum and weight decay methods and the only difference between the penalized DBN and the RODDPSO-based DBN is that the hyperparameters of the penalized DBN is chosen due to empirical experience and the hyperparameters of the RODDPSO-based DBN is determined by the RODDPSO algorithm. To summarize, the data set used in this simulation consists of 6 attributes, where 5 of them (the age, the mode of arrival, the diagnosis, the treatment time and the healthcare resource group) are inputs of the neural network, and the rest one (the attendance disposal) belongs to the output class.

The categories of the patient attendance disposal in the A&E department are illustrated in Table 6.2.

Table 6.2 Patient Attendance Disposal

Category	Description	Number of incidents
1	Admitted to a hospital bed or became a logged patient	20468
2	Discharged - follow up treatment to be provided by GP	31540
3	Discharged - did not require any follow up treatment	51795
4	Referred to A&E clinic	115
5	Referred to fracture clinic	4010
6	Referred to other outpatient clinic	7130
7	Transferred to other healthcare provider	1598
8	Died in department	62
9	Referred to other healthcare professional	5627
10	Left department before being seen for treatment	546
11	Left department having refused treatment	235
12	Other	80

Considering the suggestions from an expert in an A&E department, the number of output classes is reduced to 5. Out of all 12 real labels, 6 labels are abandoned as follows: 1) label 3, it is clear to see that the patient does not require any further follow up treatment and there may exist misdiagnosed cases, hence incidents within this category are abandoned; 2) label 4, most of the hospitals stop doing A&E clinic as they are dealing with acute emergency conditions rather than follow up cases, which could be done by GP or other outpatients specialists in recent years; 3) label 9, label 9 is part of other categories such as labels 5-7; 4) labels 10-12, these three labels are removed according to the domain knowledge of an expert in the A&E department. In addition, label 1 and label 8 are combined as one output class because the number of incidents within label 8 is very small, and patients who are admitted to a hospital bed (label 1) may have severe illness.

Overall, as displayed in Table 6.3, 5 output classes are utilized for classifying the patient attendance disposal in our work, which are: 1) admitted to a hospital bed, became a logged patient of the same health care provider or died in emergency department (20530 cases); 2) discharged, follow up treatment to be provided by the general practitioner (31540 cases); 3) referred to fracture clinic (4010 cases); 4) referred to other outpatient clinic (7130 cases); and 5) transferred to other healthcare provider (1598 cases). The diagram of the usage of the output classes is depicted in Fig. 6.6.

Table 6.3 Modified class for patient attendance disposal

Output class	Original category	Number of incidents
1	1, 8	20530
2	2	31540
3 3	5	4010
4	6	7130
5	7	1598

Data pre-processing is performed to remove the missing data and redundant attributes. Notice that the patient arrival time, the attendance conclusion date time and the treatment date time are recorded in real-time. To investigate the treatment time of each patient, the time interval between the treatment date time and the attendance conclusion date time. It is worth mentioning that the data is normalized to reduce the computation cost. In this simulation, $3,778$ incidents are deleted due to missing treatment date time or null incidents. Moreover, some attributes containing redundant information are removed during the pre-processing. For example, referral source and referral source description are used to describe the source of referral of each A&E episode where the second one is the descriptions of the first attribute. Furthermore, some irrelevant attributes such as fiscal year label and postcode sector are removed from the data set according to statistical analysis.

Parameter Setting

In the simulation, 64800 incidents are selected where 48600 incidents are used for training and the rest 16200 incidents are utilized for testing. The number of particle is 10 and the maximum iteration is set to be 10. The number of the hidden layers in the standard DBN, penalized DBN and the RODDPSO-based DBN are all set to be 3, and the number of hidden units in three hidden layers is 100, 64 and 50, respectively. The pre-training epochs of the three DBNs are all 100. The numbers of epoch of the fine-tuning process of the standard DBN, the penalized DBN, and the RODDPSO-based DBN are set to be 300. It should be mentioned that the activation function of the three variant DBNs is the sigmoid function. The parameter setting of three DBNs at the pre-training stage are given in Table 6.4. Note that we give

the range of the learning rate, the momentum, and the weight decay factor for the RODDPSO-based algorithm in Table 6.4. At the fine-tuning stage, the learning rate of the RODDPSO-based DBN is in the range of $[0, 1]$, the momentum is in the range of $[0.5, 1]$, and the weight decay factor is in set in the range of $[0, 5e-6]$.

Table 6.4 Configuration of the Standard DBN at the pre-training stage

Parameter	Standard DBN	Penalized DBN	RODDPSO-based DBN
Learning rate	0.01	0.01	[0, 0.01]
Momentum	0	0.5	[0.5, 1]
Weight decay factor	0	1e-5	[0, 1e-5]
Mini-batch size	50	50	50

Performance Metric

In our work, we aim to analyze the patient attendance disposal from emergency departments, which is a classification problem. In general, the accuracy is a fundamental performance indictor to evaluate the classification performance of the utilized classification algorithm. Here, the introduced RODDPSO-based DBN is applied to the patient attendance disposal classification problem and the standard DBN, the penalized DBN (with momentum and weight decay) and the k-nearest neighbor algorithm are employed in comparison with the RODDPSO-based DBN. The accuracy that is used as the classification performance indictor in our work is defined as follows:

$$\mathrm{A_c} = \frac{N_c}{N_c + N_f} \times 100\% \tag{6.12}$$

where N_c is the number of correct prediction, and N_f represents the number of incorrect prediction. In this case, a larger value of classification accuracy indicates a better classification performance. It should be mentioned that the mean squared error (MSE) value is calculated by the difference between the algorithm output and actual output. It is a widely used performance indicator for evaluating the performance of the trained classifiers (the standard DBN, the RODDPSO-based DBN and the penalized DBN in our work).

Experiment Results

To comprehensively evaluate the performance of the RODDPSO-based DBN, the standard DBN and the penalized DBN are used for comparison. The full-batch training MSE of the standard DBN is displayed in Fig. 6.7 where the vertical coordinate indicates the full-batch MSE and the horizontal coordinate is the epoch number. From the figure, we can see that the MSE of the DBN decreases very fast, which indicates that the pre-trained DBN performs well. In Fig. 6.8, it is apparent that the learning curve of the penalized DBN is relatively smooth. The full-batch training MSE of the DBNs which use different sets of hyperparameters (in each particle) in the final iteration are demonstrated in Fig. 6.9. The full-batch training MSE of the DBN with hyperparameters in particle 1 performs better than other DBNs with hyperparameters in other particles.

The comparison of classification accuracy of the three DBNs are depicted in Fig. 6.11. It is clear that the full-batch training MSE of the RODDPSO-based DBN is less than that of the standard DBN and the penalized DBN, which indicates that the RODDPSO-based DBN outperforms the other two DBNs. In addition, the MSE curve of the RODDPSO-based DBN decreases faster than that of the standard DBN and the penalized DBN. The classification accuracy of the RODDPSO-based DBN, the penalized DBN, and the standard DBN is 76.06%, 68.10% and 69.83%. To sum up, the RODDPSO-based DBN demonstrates superior classification performance over the penalized DBN and the standard DBN.

In addition, the classification accuracy of the KNN algorithm is depicted in Fig. 6.10. From the results, the best classification accuracy of the KNN algorithm is 75.65% when $k = 20$. By comparing the accuracy of the KNN algorithm and the RODDPSO-based DBN, we can conclude that the developed RODDPSO-based DBN demonstrates superiority over the KNN algorithm. As such, we can draw the conclusion that the RODDPSO-based DBN performs well on the patient attendance data in A&E departments. With the output class obtained by the RODDPSO-based DBN, it becomes easier to verify the patient attendance disposal category, which may improve the patient care, discharge the non-urgent patients to release the overcrowding problem and save the NHS costs in terms of both the medical and human resources.

6.5 Conclusion

In conclusion, the proposed RODDPSO algorithm has been successfully employed to improve the standard K-means clustering algorithm and fine-tune the hyperparameters in a DBN on A&E attendance data. The effectiveness of the proposed RODDPSO-based clustering algorithm is demonstrated by comparing the mean silhouette value with the K-means and FCM clustering algorithms. The developed RODDPSO-based DBN outperforms the KNN algorithm, the penalized DBN and the standard DBN on the A&E data.

Future work can be summarized into the following four aspects: (1) how to further apply the RODDPSO algorithm in other data mining problems in A&E departments and the wider health system [161, 50, 8, 67]; (2) how to use the proposed RODDPSO algorithm to improve other deep learning architectures, such as the graphical neural network and the autoencoder [189, 190]; (3) how to employ the RODDPSO algorithm in other engineering applications such as telecommunication and signal processing [166, 163, 60, 104, 103, 162]; and how to further investigate the application potential of the AWPSO algorithm (proposed in Chapter 3) and the RPSO algorithm (introduced in Chapter 4).

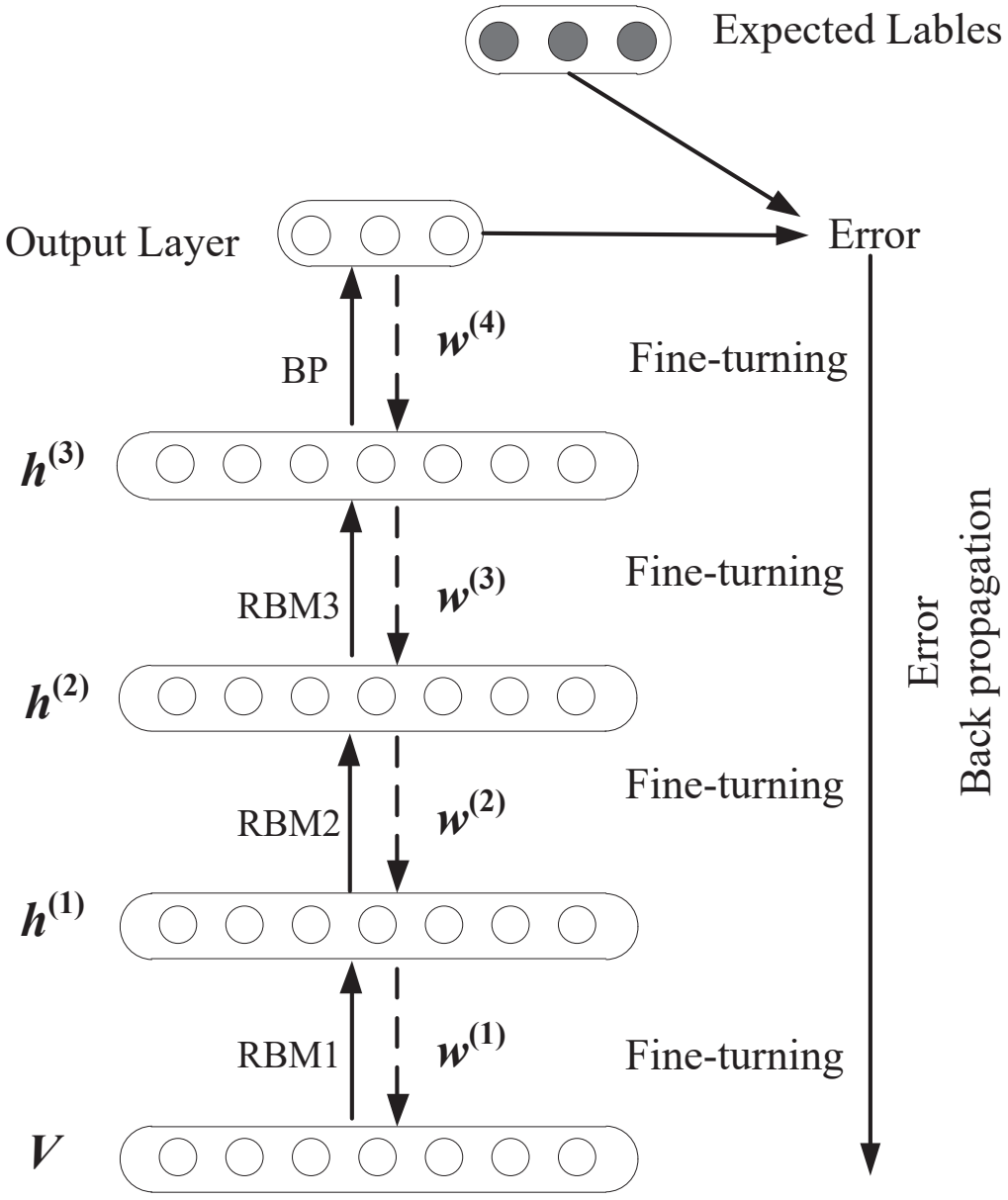

Fig. 6.5 The architecture of a DBN

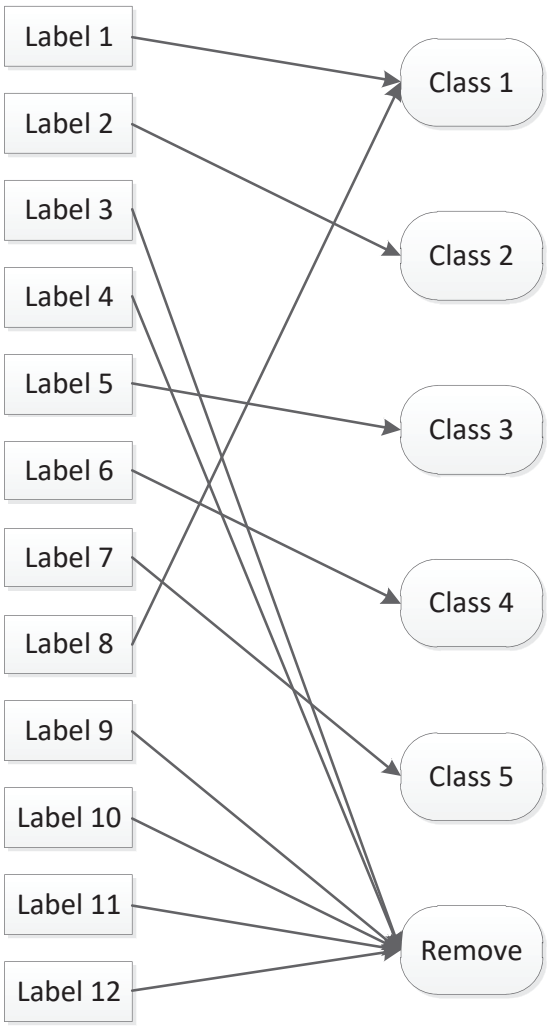

Fig. 6.6 The diagram of output classes (acquired from real labels). The detailed descriptions of each label and class are given in Table 6.2

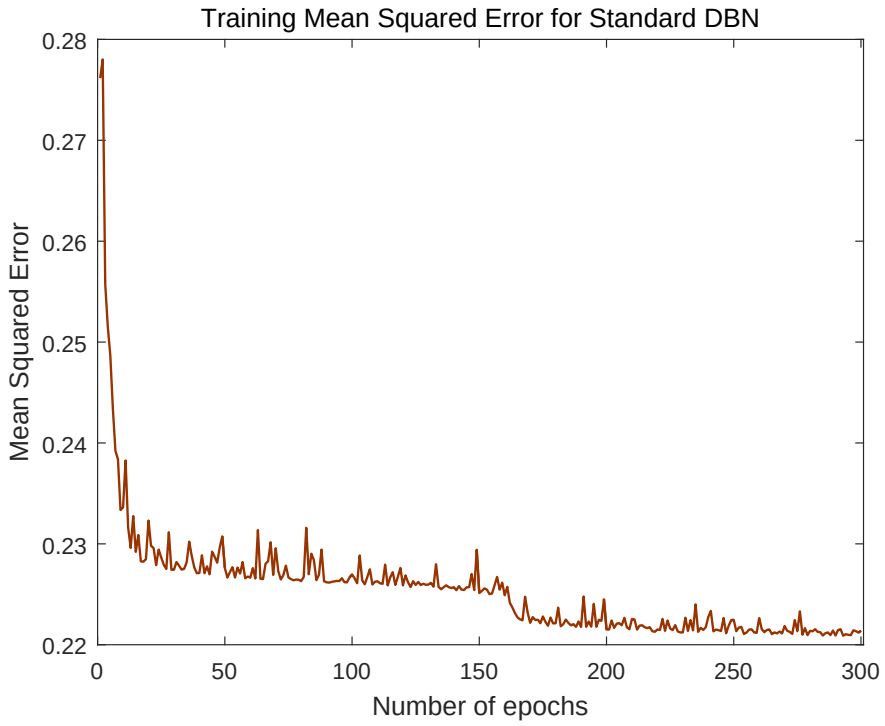

Fig. 6.7 Full-batch training mean squared error results of the DBN

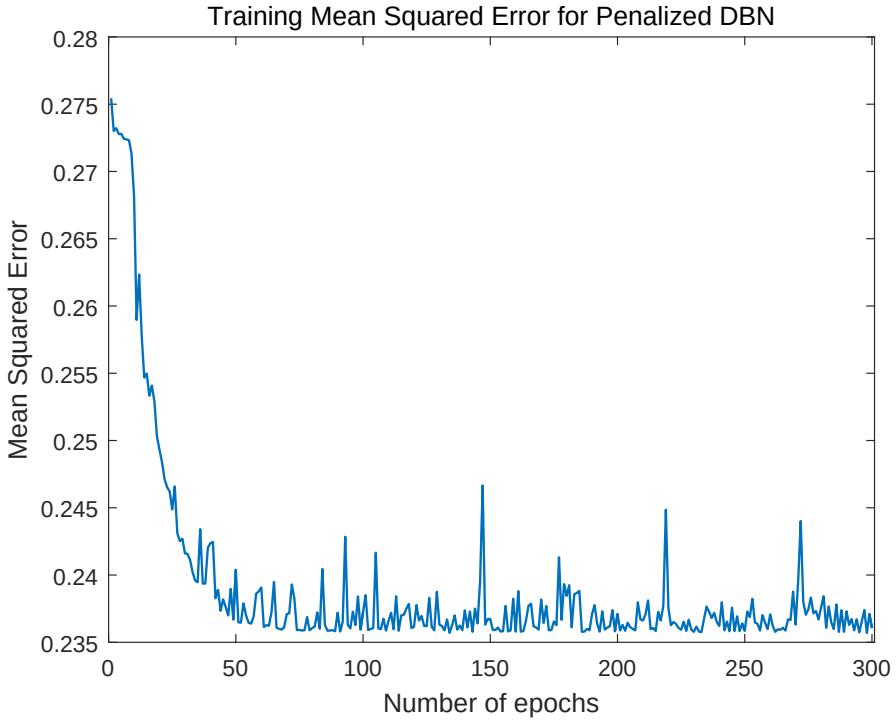

Fig. 6.8 Full-batch training mean squared error results of the DBN

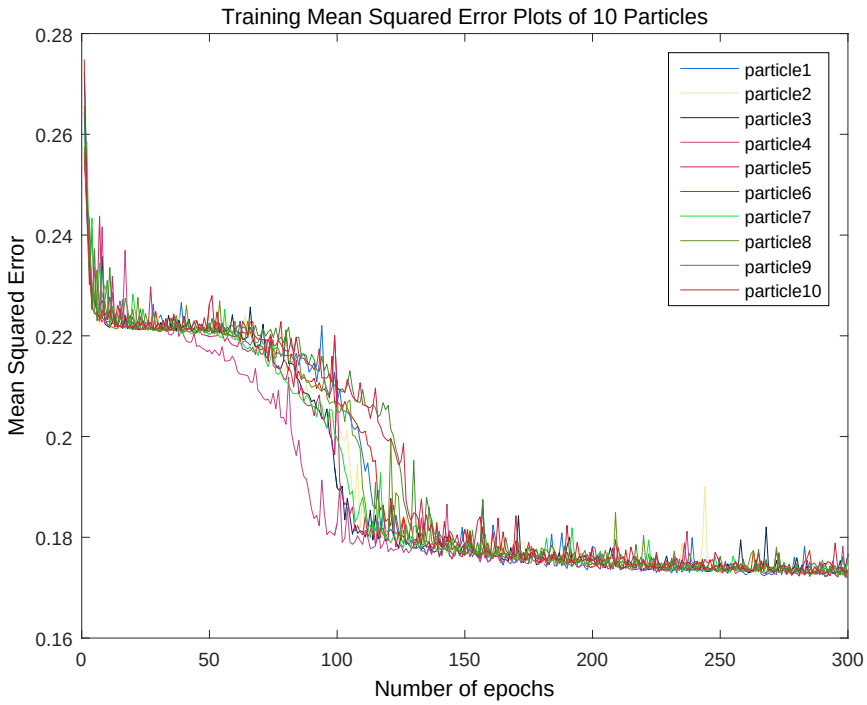

Fig. 6.9 Full-batch training mean squared error results of the DBN

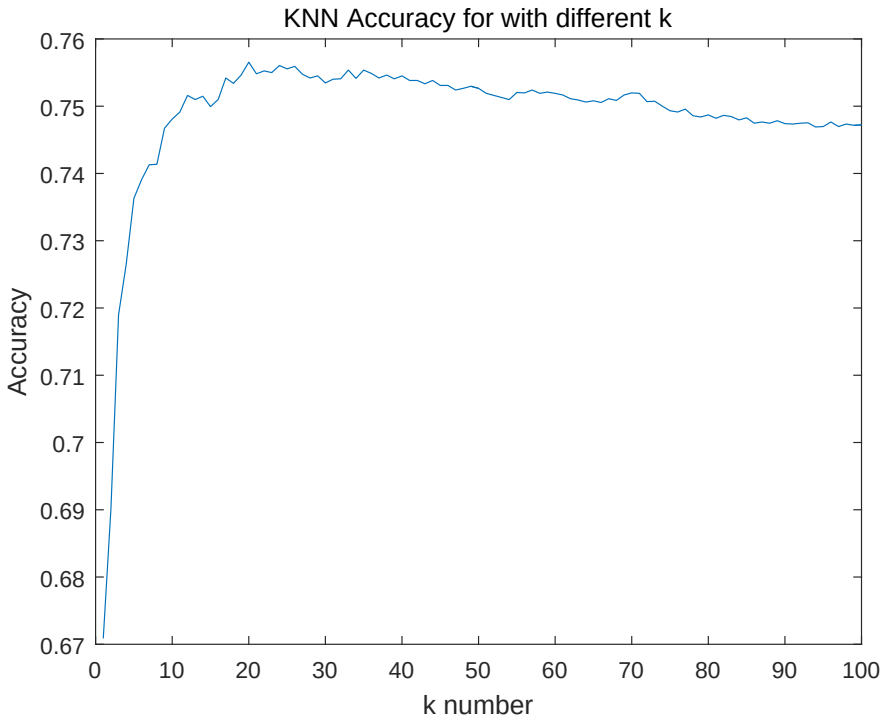

Fig. 6.10 Accuracy of the KNN algorithm when k is selected from 1 to 100

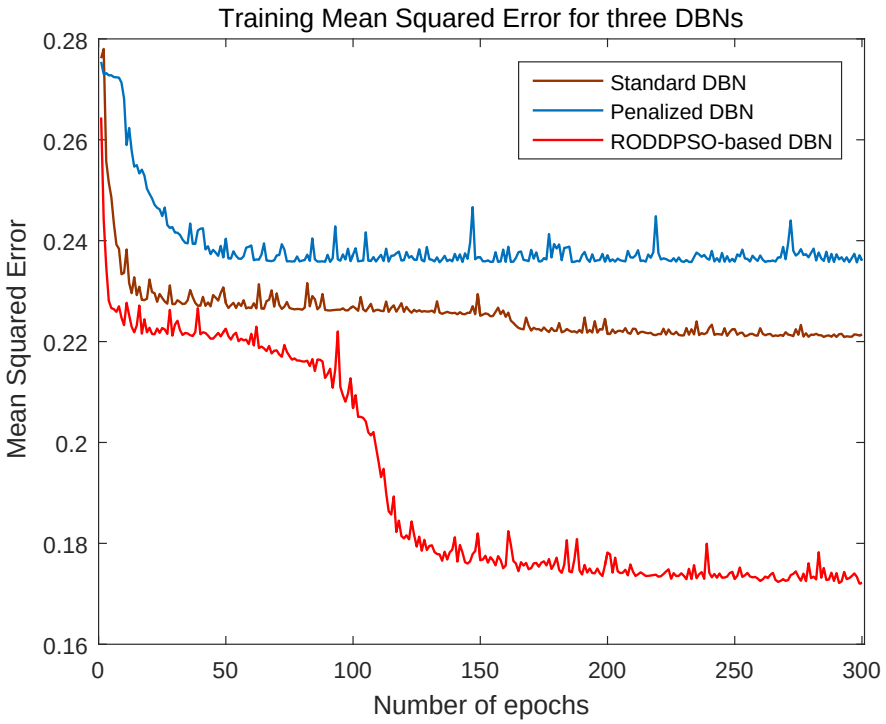

Fig. 6.11 Full-batch training mean squared error results of three DBNS (RODDPSO-based DBN, Penalized DBN, and standard DBN

Chapter 7

Conclusions and Future Research

In this chapter, we first provide a comprehensive summarization of our work in this thesis and the contributions of each chapter are also presented in Section . Then, we point out some future research directions that follow from this thesis in Section .

7.1 Summarization

In optimization problems, the problem of premature convergence poses great challenges to evolutionary computation algorithms, and the PSO algorithm is not an exception. The balance between the local exploitation and the global exploration plays a significant role in discovering the optimal solution and affects the convergence rate of the optimizer. In this light, we aim to put forward new PSO algorithms by designing advanced parameter updating schemes and novel topological structures of the velocity updating model in order to maintain the population diversity and alleviate the premature convergence problem.

In this thesis, we have proposed some variant PSO algorithms with the purpose of further improving the search capability of the optimizer in terms of the population diversity and the convergence rate so as to alleviate the premature convergence problem. An adaptive weighting PSO (AWPSO) algorithm has been designed in order to improve the convergence rate of the optimizer (Chapter 3). A randomized PSO (RPSO) algorithm has been proposed where a sigmoid-function-based weighting mechanism has been designed to randomly perturb the acceleration coefficients with hope to

thoroughly exploit and explore the problem space (Chapter 4). A novel randomly occurring distributedly delayed PSO (RODDPSO) algorithm has been introduced which could not only thoroughly seek the optimal solution through the entire search space but also reduce the possibility of stagnating within local optima (Chapter 5). The application potential of the PSO algorithm has been addressed, and the proposed RODDPSO algorithm has been successfully applied to data clustering and deep learning applications with regards to A&E departments (Chapter 6). In the following paragraphs, the research outputs in each chapter are concluded.

In Chapter 3, a novel AWPSO algorithm has been proposed with hope to improve the convergence rate of the standard particle swarm optimizer. The acceleration coefficients of the AWPSO algorithm are adaptively adjusted according to the designed sigmoid-function-based weighting mechanism. The designed weighting mechanism employs a sigmoid function and the distances from the particle to its *pbest* and from the particle to the *gbest* are both taken into consideration, thereby contributing to the enhancement of the convergence rate of the optimizer. It has been observed that the AWPSO algorithm has shown competitive performance by comparing to several popular PSO algorithms in terms of the convergence rate via eight widely used optimization benchmark functions.

In Chapter 4, a RPSO algorithm has been presented in order to enhance the search capability of the PSO algorithm where the Gaussian white noises (GWNs) have been introduced to randomly perturb the social and cognitive acceleration coefficients. Under this parameter updating strategy, the RPSO algorithm could explore and exploit the search space more thoroughly, thereby having the distinguishing feature of easily escaping the local optima trap. Experimental results have demonstrated that the devised RPSO algorithm outperforms six existing PSO algorithms via eight popular optimization benchmark functions.

In Chapter 5, a novel RODDPSO algorithm has been developed by introducing the distributed time-delay (DTD) terms into the velocity updating model. The velocity updating model of the RODDPSO algorithm is adaptively adjusted depending on the evolutionary state. The DTD terms make full use of historical information during the evolution process and contribute to a thorough exploration of the problem space. In

this case, the convergence behaviors of the optimizer are improved and the capability of escaping the local optima is enhanced. The superiority of the proposed RODDPSO algorithm is demonstrated over six popular variant PSO algorithms via a series of optimization benchmark functions including both the unimodal and multimodal cases.

In Chapter 6, the intelligent data analysis applications of the PSO algorithm in the healthcare domain have been discussed. The developed RODDPSO algorithm has been successfully employed for intelligent data analysis of the patient attendance data in an A&E department in West London. The detailed information of the A&E data has been presented, and two practical data mining problems (the patient clustering problem and the patient classification problem) have been studied on the A&E data. First, RODDPSO algorithm has been exploited in analyzing the A&E data by improving the traditional K-means clustering algorithm with the purpose of generating an accurate triage category for the patients who attend the A&E departments. The RODDPSO-based clustering algorithm has shown promising performance for data clustering as the cluster centroids are obtained by using the RODDPSO algorithm instead of randomly selecting the cluster centroids. Experiment results have illustrated that the RODDPSO-based clustering method outperforms two other well-known clustering algorithms. Second, the RODDPSO algorithm has been applied to the popular deep learning techniques by optimizing the hyperparameters of the deep belief network (DBN). The developed RODDPSO-based DBN has been successfully applied to analyze the A&E data for predicting the patient attendance disposal in a London A&E department.

7.2 Future Work

In this thesis, we have studied the well-known evolutionary computation approach, the PSO algorithm. To address the challenging problems of the PSO algorithm, three variant PSO algorithms have been proposed with hope to improve the search capability of the PSO algorithms in terms of the population diversity and the convergence. Although the developed approaches have been successfully applied to intelligent data analysis, there is still much room to extend our work from perspectives of algorithm

and application. Below are some future research directions relevant to the research work in this thesis.

- In this thesis, the effectiveness of the proposed three novel PSO algorithms is evaluated by comparing with that of some existing PSO algorithms on the widely used benchmark functions. Some research work has been focused on the convergence of the PSO algorithms [17, 27, 202]. In the future, we aim to perform a thorough empirical investigation of the stability and convergence of the proposed PSO algorithms.

- The initialization of the population size, the position and the velocity of the particles is a challenging problem for dealing with high-dimensional problems. In addition, another challenging problem is to discover the optimal solution efficiently and accurately. Notice that there aren't any PSO algorithms capable of fully solving the premature convergence problem, especially in the high-dimensional search space. Under this circumstance, one future research direction is to consider the initialization process and develop new variant PSO algorithms with hope to alleviate premature convergence.

- It is well known that in the standard PSO algorithm, all the particles update their velocity and position based on their personal best position and the global best position. However, some recent results demonstrate that the PSO algorithms without considering the global best position may also perform well and even outperform the standard PSO algorithm. In this case, the memory of the particles should attract some interest. It is possible to imagine that the current particle could interact with its neighbors' memories and then get updated. Under this circumstance, the typology of the neighbors should be investigated and considered as another future research direction.

- In this thesis, we only apply the RODDPSO algorithm proposed in Chapter 5 to intelligent data analysis. In the near future, we aim to apply our other algorithms to other practical applications, such as path planning [140], system engineering and signal processing [157, 185, 20]. Also, how to extend our results to other health informatics arises to be an interesting future topic [161].

- The multi-objective or many-objective optimization are also important research directions in the optimization community. The multi-objective optimization problem has attracted increasingly interest in various areas, e.g., healthcare informatics, signal processing, power systems, and manufacturing [58, 174, 69, 184, 99, 202, 147]. In this context, we aim to extend our results to the multi-objective and many-objective optimization problems.

- In this thesis, the effectiveness of the proposed three novel PSO algorithms is evaluated by comparing with that of some existing PSO algorithms via the widely used benchmark functions. Some theoretical research work has been focused on the convergence of the PSO algorithms [17, 27, 202]. In the future, we aim to perform a thorough empirical investigation of the stability and convergence of the proposed PSO algorithms.

References

[1] O. Abdel-Hamid, A. R. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 10, pp. 1533–1545, 2014.

[2] A. H. Abdi, C. Luong, T. Tsang, G. Allan, S. Nouranian, J. Jue, D. Hawley, S. Fleming, K. Gin, J. Swift, and R. Rohling, "Automatic quality assessment of echocardiograms using convolutional neural networks: feasibility on the apical four-chamber view," *IEEE Transactions on Medical Imaging*, vol. 36, no. 6, pp. 1221–1230, 2017.

[3] Centre for Change and Innovation, "Decide to admit v admit to decide - working together to improve unscheduled care in NHS scotland," *National Conference Unplanned Care for Medical Conditions*, 2004.

[4] M. Afilal, F. Yalaoui, F. Dugardin, L. Amodeo, D. Laplanche, and P. Blua, "Forecasting the emergency department patients flow," *Journal of Medical Systems*, vol. 40, pp. 1–18, 2016.

[5] P. Agarwalla, and S. Mukhopadhyay, "Efficient player selection strategy based diversified particle swarm optimization algorithm for global optimization", *Information Sciences*, vol. 397, pp. 69–90, 2017.

[6] M. Alswaitti, M. Albughdadi, and N. A. M. Isa, "Density-based particle swarm optimization algorithm for data clustering," *Expert Systems with Applications*, vol. 91, pp. 170–186, 2018.

[7] M. R. AlRashidi, and M. E. El-Hawary, "A survey of particle swarm optimization applications in electric power systems", *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 913–918, 2009.

[8] A. Azadeh, M. H. Farahani, S. Torabzadeh, and M. Baghersad, "Scheduling prioritized patients in emergency department laboratories," *Computer Methods and Programs in Biomedicine*, vol. 117, no. 2, pp. 61–70, 2014.

[9] W. Bai, I. Eke, and K. Y. Lee, "An improved artificial bee colony optimization algorithm based on orthogonal learning for optimal power flow problem," *Control Engineering Practice*, vol. 61, pp. 163–172, 2017.

[10] R. B. Balaban, J. S. Weissman, P. A. Samuel, and S. Woolhandler, "Redefining and redesigning hospital discharge to enhance patient care: a randomized controlled study," *Journal of General Internal Medicine*, vol. 23, no. 8, pp. 1228–1233, 2008.

[11] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layerwise training of deep networks," In: *Proceedings of Advances in Neural Information Processing Systems 19*, Vancouver, BC, Canada, Dec. 2006, pp. 153–160.

[12] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[13] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kegl, "Algorithms for hyper-parameter optimization," In: *Proceedings of Advances in Neural Information Processing Systems 24*, Vancouver, BC, Canada, Dec. 2011, pp. 2546–2554.

[14] M. Berthold, D. J. Hand, *Intelligent Data Analysis: An Introduction*, Springer-Verlag New York, Inc, 1999.

[15] P. Bhattacharjee and P. K. Ray, "Patient flow modelling and performance analysis of healthcare delivery processes in hospitals: a review and reflections," *Computers and Industrial Engineering*, vol. 78, pp. 299–312, 2014.

[16] N. Bobrovitz, D. S. Lasserson, and A. D. Briggs, "Who breaches the four-hour emergency department wait time target? A retrospective analysis of 374,000 emergency department attendances between 2008 and 2013 at a type 1 emergency department in England," *BMC Emergency Medicine*, vol. 17, no. 1, pp. 32, 2017.

[17] M. R. Bonyadi and Z. Michalewicz, "Analysis of stability, local convergence, and transformation sensitivity of a variant of the particle swarm optimization algorithm", *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 3, pp. 370–385, 2016.

[18] M. R. Bonyadi, and Z. Michalewicz, "Impacts of coefficients on movement patterns in the particle swarm optimization algorithm", *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 3, pp. 378–390, 2017.

[19] L. Cao, L. Xu, and E. D. Goodman, "A collaboration-based particle swarm optimizer with history-guided estimation for optimization in dynamic environments," *Expert Systems with Applications*, vol. 120, pp. 1–13, 2019.

[20] F. Chen, L. Wang, B. Jiang, and C. Wen, "An arterial traffic signal control system based on a novel intersections model and improved hill climbing algorithm," *Cognitive Computation*, vol. 7, no. 4, pp. 464–476, 2015.

[21] J. Chen, J. Zheng, P. Wu, L. Zhang, and Q. Wu, "Dynamic particle swarm optimizer with escaping prey for solving constrained non-convex and piecewise optimization problems", *Expert Systems with Applications*, vol. 86, pp. 208–223, 2017.

[22] K. Chen, F. Zhou, L. Yin, S. Wang, Y. Wang, and F. Wan, "A hybrid particle swarm optimizer with sine cosine acceleration coefficients", *Information Sciences*, vol. 422, pp. 218–241, 2018.

[23] W.-N. Chen, J. Zhang, Y. Lin, N. Chen, Z.-H. Zhan, H. S.-H. Chung, Y. Li and Y.-H. Shi, "Particle swarm optimization with an aging leader and challengers,"

IEEE Transactions on Evolutionary Computation, vol. 17, no. 2, pp. 241-258, 2013.

[24] Y. Chen and Y. Yao, "A multiview approach for intelligent data analysis based on data operators," *Information Sciences*, vol. 178, no. 1, pp. 1–20, 2008.

[25] R. Cheng and Y. Jin, "A competitive swarm optimizer for large scale optimization," *IEEE Transactions on Cybernetics*, vol. 45, no. 2, pp. 191–204, 2015.

[26] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Deep, big, simple neural nets for handwritten digit recognition," *Neural Computation*, vol. 22, no. 12, pp. 3207–3220, 2010.

[27] M. Clerc and J. Kennedy, "The particle swarm: explosion, stability, and convergence in a multi-dimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[28] T. J. Coats and S. Michalis, "Mathematical modelling of patient flow through an accident and emergency department," *Emergency Medicine Journal*, vol. 18, no. 3, pp. 190–192, 2001.

[29] R. Collobert, and J. Weston, "A unified architecture for natural language processing: deep neural networks with multitask learning," In: *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, Finland, Jul. 2008, pp. 160–167.

[30] M. Crepinsek, S. H. Liu, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: a survey," *ACM Computing Surveys*, vol. 45, no. 3, pp. 1–33, 2013.

[31] Q. Cui, Q. Li, G. Li, Z. Li, X. Han, H. P. Lee, Y. Liang, B. Wang, J. Jiang, and C. Wu, "Globally-optimal prediction-based adaptive mutation particle swarm optimization", *Information Sciences*, vol. 418, pp. 186–217, 2017.

[32] S. Das, A. Abraham, and A. Konar, "Automatic kernel clustering with a multi-elitist particle swarm optimization algorithm," *Pattern Recognition Letters*, vol. 29, no. 5, pp. 688–699, 2008.

[33] 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.

[34] R. P. De Gusmao, and F. D. A. De Carvalho, "Clustering of multi-view relational data based on particle swarm optimization," *Expert Systems with Applications*, vol. 123, pp. 34–53, 2019.

[35] Y. Del Valle, G. K. Venayagamoorthy, S. Mohagheghi, J. C. Hernandez, and R. G. Harley, "Particle swarm optimization: basic concepts, variants and applications in power systems," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 2, pp. 171–195, 2008.

[36] G. H. De Rosa, J. P. Papa, and X. S. Yang, "Handling dropout probability estimation in convolution neural networks using meta-heuristics," *Soft Computing*, vol. 22, no. 18, pp. 6147–6156, 2018.

[37] L. Deng and D. Yu, "Deep learning: methods and applications," *Foundations and Trends in Signal Processing*, vol. 7, pp. 197–387, 2014.

[38] T. Desautels, R. Das, J. Calvert, M. Trivedi, C. Summers, D. J. Wales, and A. Ercole, "Prediction of early unplanned intensive care unit readmission in a UK tertiary care hospital: a cross-sectional machine learning approach," *BMJ Open*, vol. 7, article no. e017199, 2017.

[39] I. S. Dhillon and D. S. Modha, "Concept decompositions for large sparse text data using clustering," *Machine Learning*, vol. 42, no. 1, pp. 143–175, 2001.

[40] K. A. De Jong, *Evolutionary Computation: A Unified Approach*, Cambridge, MA: MIT Press, 2006.

[41] A. C. Durand, S. Gentile, B. Devictor, S. Palazzolo, P. Vignally, P. Gerbeaux, and R. Sambuc, "ED patients: how nonurgent are they? Systematic review of the emergency medicine literature," *The American Journal of Emergency Medicine*, vol. 29, no. 3, pp. 333–345, 2011.

[42] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295–307, 2015.

[43] W. Dong and M. C. Zhou, "A supervised learning and control method to improve particle swarm optimization algorithms," *IEEE Transactions on Systems, Man, and Cybernetics–Part A: Systems*, vol. 47, no. 7, pp. 1135-1148, 2017.

[44] C. Du, Z. Yin, Y. Zhang, J. Liu, X. Sun, and Y. Zhong, "Research on active disturbance rejection control with parameter autotune mechanism for induction motors based on adaptive particle swarm optimization algorithm with dynamic inertia weight," *IEEE Transactions on Power Electronics*, vol. 34, no. 3, pp. 2841–2855, 2019.

[45] J. Eatock, M. Clarke, C. Picton, and T. Young, "Meeting the four-hour deadline in an A&E department," *Journal of Health Organization and Management*, vol. 25, no. 6, pp. 606–624, 2011.

[46] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," In: *Proceedings of the 6th International Symposium on Micro Machine and Human Science*, Nagoya, Japan, Oct. 1995, pp. 39–43.

[47] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," In: *Proceedings of the 2000 Congress on Evolutionary Computation*, San Diego, USA, Jul. 2000, pp. 84–88.

[48] R. C. Eberhart and Y. Shi, "Particle swarm optimization: developments, applications and resources," In: *Proceedings of the 2001 Congress on Evolutionary Computation*, Seoul, South Korea, May 2001, vol. 1, pp. 81–86.

[49] M. Elhoseny, A. Abdelaziz, A. S. Salama, A. M. Riad, K. Muhammad, and A. K. Sangaiah, "A hybrid model of internet of things and cloud computing to manage big data in health services applications," *Future Generation Computer Systems*, vol. 86, pp. 1383–1394, 2018.

[50] M. P. Fanti, A. M. Mangini, M. Dotoli, and W. Ukovich, "A three-level strategy for the design and performance evaluation of hospital departments," *IEEE Transactions on Systems, Man, and Cybernetics–Part A: Systems*, vol. 43, no. 4, pp. 742–756, 2013.

[51] W. A. Feess and S. G. Stephens, "Evaluation of GPS ionospheric time-delay model," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 3, pp. 332–338, 1987.

[52] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[53] S. Fujino, T. Hatanaka, N. Mori, and K. Matsumoto, "Evolutionary deep learning based on deep convolutional neural network for anime storyboard recognition," *Neurocomputing*, vol. 338, pp. 393–398, 2019.

[54] A. GaLvez and A. Iglesias, "A new iterative mutually coupled hybrid GA-PSO approach for curve fitting in manufacturing," *Applied Soft Computing*, vol. 13, no. 3, pp. 1491–1504, 2013.

[55] M. Gan, and C. Wang, "Construction of hierarchical diagnosis network based on deep learning and its application in the fault pattern recognition of rolling element bearings," *Mechanical Systems and Signal Processing*, vol. 72, pp. 92–104, 2016.

[56] A. H. Gandomi, K. Deb, R. C. Averill, S. Rahnamayan, and M. N. Omidvar, "Using semi-independent variables to enhance optimization search. Expert Systems with Applications, vol. 120, pp. 279–297, 2019.

[57] H. Garg, "A hybrid PSO-GA algorithm for constrained optimization problems," *Applied Mathematics and Computation*, vol. 274, pp. 292–305, 2016.

[58] J. Garcia-Nieto, E. Lopez-Camacho, M. J. Garcia-Godoy, A. J. Nebro, and J. F. Aldana-Montes, "Multi-objective ligand-protein docking with particle swarm optimizers," *Swarm and Evolutionary Computation*, vol. 44, pp. 439–452, 2019.

[59] A. Garcia-Villoria and R. Pastor, "Introducing dynamic diversity into a discrete particle swarm optimization," *Computers & Operations Research*, vol. 36, no. 3, pp. 951–966, 2009.

[60] H. Geng, Y. Liang, Y. Liu, and F. E. Alsaadi, "Bias estimation for asynchronous multi-rate sensor fusion with unknown inputs," *Information Fusion*, vol. 39, pp. 139–153, 2018.

[61] P. Ghamisi and J. A. Benediktsson, "Feature selection based on hybridization of genetic algorithm and particle swarm optimization," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 2, pp. 309–313, 2015.

[62] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: a deep learning approach," In: *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, Washington, USA, Jun. 2011, pp. 513–520.

[63] M. Gong, Q. Cai, X. Chen, and L. Ma, "Complex network clustering by multiobjective discrete particle swarm optimization based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 1, pp. 82–97, 2014.

[64] M. Gong, J. Liu, H. Li, Q. Cai, and L. Su, "A multiobjective sparse feature learning model for deep neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 12, pp. 3263–3277, 2015.

[65] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, Cambridge: MIT press, 2016.

[66] N. Goyette, P. M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection.net: A new change detection benchmark dataset," In: *Proceedings of the 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, Providence, USA, Jun. 2012.

[67] M. Gul and A. F. Guneri, "A comprehensive review of emergency department simulation applications for normal and disaster conditions," *Computers & Industrial Engineering*, vol. 83, pp. 327–344, 2015.

[68] P. Guo, W. Feng, M. Zheng, J. Lv, L. Wang, J. Liu, Y. Zhang, G. Luo, Y. Zhang, C. Deng, and T. Shi, "Short-term associations of ambient air pollution and cause-specific emergency department visits in Guangzhou, China," *Science of the Total Environment*, vol. 613, pp. 306–313, 2018.

[69] H. Han, W. Lu, L. Zhang, and J. Qiao, "Adaptive gradient multiobjective particle swarm optimization," *IEEE Transactions on Cybernetics*, vol. 99, pp. 3067–3079, 2017.

[70] D. J. Hand, "Intelligent data analysis: issues and opportunities," *Intelligent Data Analysis*, vol. 2, pp. 67–79, 1998.

[71] M. He, M. Liu, R. Wang, X. Jiang, B. Liu, and H. Zhou, "Particle swarm optimization with damping factor and cooperative mechanism," *Applied Soft Computing*, vol. 76, pp. 45–52, 2019.

[72] G. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[73] G. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[74] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[75] F. Hu, G. S. Xia, J. Hu, and L. Zhang, "Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery," *Remote Sensing*, vol. 7, no. 11, pp. 14680–14707, 2015.

[76] M. Hu, T. Wu, and J. D. Weir, "An adaptive particle swarm optimization with multiple adaptive methods," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 5, pp. 705–720, 2013.

[77] H. Izakian and A. Abraham, "Fuzzy C-means and fuzzy swarm for fuzzy clustering problem," *Expert Systems with Applications*, vol. 38, no. 3, pp. 1835–1838, 2011.

[78] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[79] Y.-H. Jia, W.-N. Chen, T. Gu, H. Zhang, H. Yuan, Y. Lin, W.-J. Yu and J. Zhang, A dynamic logistic dispatching system with set-based particle swarm optimization, *IEEE Transactions on Systems, Man, and Cybernetics–Part A: Systems*, vol. 48, no. 9, pp. 1607-1621, Sept. 2018.

[80] S. Jiang, K.-S. Chin, L. Wang, G. Qu, and K. L. Tsui, "Modified genetic algorithm-based feature selection combined with pre-trained deep neural network for demand forecasting in outpatient department," *Expert Systems with Applications*, vol. 82, pp. 216–230, 2017.

[81] S. Jiang, K.-S. Chin, and K. L. Tsui, "A universal deep learning approach for modeling the flow of patients under different severities," *Computer Methods and Programs in Biomedicine*, vol. 154, pp. 191–203, 2018.

[82] J. Kennedy, and R. C. Eberhart, "Particle swarm optimization," In: *Proceedings of the 1995 IEEE International Conference on Neural Networks*, Perth, Australia, Nov. 1995, pp. 1942–1948.

[83] I. J. Kim, and X. Xie, "Handwritten hangul recognition using deep convolutional neural networks," *International Journal on Document Analysis and Recognition*, vol. 18, no. 1, pp. 1–13, 2015.

[84] K. Krishna and M. N. Murty, "Genetic K-means algorithm," *IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics*, vol. 29, no. 3, pp. 433–439, 1999.

[85] P. Landa, M. Sonnessa, E. Tanfani, and A. Testi, "Multiobjective bed management considering emergency and elective patient flows," *International Transactions in Operational Research*, vol. 25, no. 1, pp. 91–110, 2018.

[86] N. D. Lane, and P. Georgiev, "Can deep learning revolutionize mobile sensing?" In: *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, Santa Fe, USA, Feb. 2015, pp. 117–122.

[87] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[88] H. Lee, P. Pham, Y. Largman, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," In: *Proceedings of Advances in Neural Information Processing Systems 22*, Vancouver, B.C., Canada, Dec. 2009, pp. 1096–1104.

[89] M. Li, S. Yang, and X. Liu, "A performance comparison indicator for Pareto front approximations in many-objective optimization," In: *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, Madrid, Spain, Jul. 2015, pp. 703–710.

[90] X. Li, "Niching without niching parameters: particle swarm optimization using a ring topology", *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 150–169, 2010.

[91] X. Li and X. Yao, "Cooperatively coevolving particle swarms for large scale optimization," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 2, pp. 210–224, 2012.

[92] S.-W. Lin, K.-C. Ying, S.-C. Chen, and Z.-J. Lee, "Particle swarm optimization for parameter determination and feature selection of support vector machines", *Expert Systems with Applications*, vol. 35, no. 4, pp. 1817–1824, 2008.

[93] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

[94] C. L. Liao, S. J. Lee, Y. S. Chiou, C. R. Lee, and C. H. Lee, "Power consumption minimization by distributive particle swarm optimization for luminance control and its parallel implementations", *Expert Systems with Applications*, vol. 96, pp. 479–491, 2018.

[95] Q. Lin, S. Liu, Q. Zhu, C. Tang, R. Song, J. Chen, C. A. C. Coello, K. C. Wong, and J. Zhang, "Particle swarm optimization with a balanceable fitness estimation for many-objective optimization problems", *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 32–46, 2018.

[96] Z. H. Ling, S. Y. Kang, H. Zen, A. Senior, M. Schuster, X. J. Qian, H. M. Meng, and L. Deng, "Deep learning for acoustic modeling in parametric speech generation: a systematic review of existing techniques and future trends," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 35–52, 2015.

[97] H.-R. Liu, J.-C. Cui, Z.-D. Lu, D.-Y. Liu, and Y.-J. Deng, "A hierarchical simple particle swarm optimization with mean dimensional information," *Applied Soft Computing*, vol. 76, pp. 712–725, 2019.

[98] J. Liu, Y. Mei, and X. Li, "An analysis of the inertia weight parameter for binary particle swarm optimization", *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 666–681, 2016.

[99] J. Liu, H. Zhang, K. He, and S. Jiang, "Multi-objective particle swarm optimization algorithm based on objective space division for the unequal-area facility layout problem", *Expert Systems with Applications*, vol. 102, pp. 179–192, 2018.

[100] N. Liu, Z. X. Koh, E. C. P. Chua, L. M. L. Tan, Z. Lin, B. Mirza, and M. E. H. Ong, "Risk scoring for prediction of acute cardiac complications from imbalanced clinical data," *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 6, pp. 1894–1902, 2014.

[101] Q. Liu, W. Wei, H. Yuan, Z.-H. Zhan, and Y. Li, "Topology selection for particle swarm optimization", *Information Sciences*, vol. 363, pp. 154–173, 2016.

[102] R. Liu, L. Ma, Y. Wang, and L. Zhang, "Learning converged propagations with deep prior ensemble for image enhancement," *IEEE Transactions on Image Processing*, vol. 28, no. 3, pp. 1528–1543, 2019.

[103] S. Liu, Z. Wang, L. Wang, and G. Wei, "On quantized H_∞ filtering for multi-rate systems under stochastic communication protocols: the finite-horizon case," *Information Sciences*, vol. 459, pp. 211–223, 2018.

[104] S. Liu, Z. Wang, G. Wei, and M. Li, "Distributed set-membership filtering for multirate systems under the Round-Robin scheduling over sensor networks," *IEEE Transactions on Cybernetics*, doi:10.1109/TCYB.2018.2885653.

[105] W. Liu, Z. Wang, X. Liu, N. Zeng, and D. Bell, "A novel particle swarm optimization approach for patient clustering from emergency departments," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 4, pp. 632–644, 2019.

[106] Z. G. Liu, X. H. Ji, and Y. X. Liu, "Hybrid non-parametric particle swarm optimization and its stability analysis", *Expert Systems with Applications*, vol. 92, pp. 256–275, 2018.

[107] Q. Lu, Q.-L. Han, and S. Liu, "A finite-time particle swarm optimization algorithm for odor source localization," *Information Sciences*, vol. 277, pp. 111–140, 2014.

[108] Q. Lu, Q.-L. Han, and S. Liu, "A cooperative control framework for a collective decision on movement behaviors of particles," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 6, pp. 859–873, 2016.

[109] N. Lynn, M. Z. Ali, and P. N. Suganthan, "Population topologies for particle swarm optimization and differential evolution", *Swarm and Evolutionary Computation*, vol. 39, pp. 24–35, 2018.

[110] R. Manor, and A. B. Geva, "Convolutional neural network for multi-category rapid serial visual presentation BCI," *Frontiers in computational neuroscience*, vol. 9, 2015.

[111] Y. Marinakis, M. Marinaki, and A. Migdalas, "A multi-adaptive particle swarm optimization for the vehicle routing problem with time windows," *Information Sciences*, vol. 481, pp. 311–329, 2019.

[112] J. Marrow, "Triage and casemix in accident and emergency medicine," *European Journal of Emergency Medicine*, vol. 5, no. 1, pp. 53–58, 1998.

[113] M. Mavrovouniotis, C. Li, and S. Yang, "A survey of swarm intelligence for dynamic optimization: algorithms and applications", *Swarm and Evolutionary Computation*, vol. 33, pp. 1–17, 2017.

[114] J. Matos, R. P. Faria, I. B. Nogueira, J. M. Loureiro, and A. M. Ribeiro, "Optimization strategies for chiral separation by true moving bed chromatography using particles swarm optimization (PSO) and new parallel PSO variant," *Computers & Chemical Engineering*, vol. 123, pp. 344-356, 2019.

[115] A. R. Mohamed, G. Dahl, and G. Hinton, "Deep belief networks for phone recognition," In: *Proceedings of NIPS Workshop on Deep Learning for Speech Recognition and Related Applications*, Dec. 2009.

[116] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, New York, 1995.

[117] E. Moy, R. M. Coffey, B. J. Moore, M. L. Barrett, and K. K. Hall, "Length of stay in EDs: variation across classifications of clinical condition and patient discharge disposition," *The American Journal of Emergency Medicine*, vol. 34, no. 1, pp. 83–87, 2016.

[118] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *Journal of Big Data*, vol. 2, no. 1, pp. 1–21, 2015.

[119] T. Niknam and B. Amiri, "An efficient hybrid approach based on PSO, ACO and K-means for cluster analysis," *Applied Soft Computing*, vol. 10, no. 1, pp. 183–197, 2010.

[120] K. Noda, Y. Yamaguchi, K. Nakadai, H. G. Okuno, and T. Ogata, "Audio-visual speech recognition using deep learning," *Applied Intelligence*, vol. 42, no. 4, pp. 722–737, 2015.

[121] C. Ozgun-Kibiroglu, M. N. Serarslan, and Y. I.Topcu, "Particle swarm optimization for uncapacitated multiple allocation hub location problem under congestion," *Expert Systems with Applications*, vol. 119, pp. 1–19, 2019.

[122] A. Pan, L. Wang, W. Guo, and Q. Wu, "A diversity enhanced multiobjective particle swarm optimization", *Information Sciences*, vol. 436, pp. 441–465, 2018.

[123] J. P. Papa, W. Scheirer, and D. D. Cox, "Fine-tuning deep belief networks using harmony search," *Applied Soft Computing*, Vol. 46, pp. 875-885, 2016.

[124] J. P. Papa, G. H. Rosa, D. R. Pereira, and X. S. Yang, "Quaternion-based deep belief networks fine-tuning," *Applied Soft Computing*, vol. 60, pp. 328–335, 2017.

[125] J. B. Park, Y. W. Jeong, J. R. Shin, and K. Y. Lee, "An improved particle swarm optimization for nonconvex economic dispatch problems", *IEEE Transactions on Power Systems*, vol. 25, no. 1, pp. 156–166, 2010.

[126] L. A. Passos, D. R. Rodrigues, and J. P. Papa, "Fine tuning deep boltzmann machines through meta-heuristic approaches," In: *Proceedings of the IEEE 12th International Symposium on Applied Computational Intelligence and Informatics*, Timisoara, Romania, May 2018, pp. 419–424.

[127] Q. Qin, S. Cheng, Q. Zhang, L. Li, and Y. Shi, "Particle swarm optimization with interswarm interactive learning strategy", *IEEE Transactions on Cybernetics*, vol. 46, no. 10, pp. 2238–2251, 2016.

[128] C. C. Queirolo, L. Silva, O. R. Bellon, and M. P. Segundo, "3D face recognition using simulated annealing and the surface interpenetration measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 2, pp. 206–219, 2010.

[129] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.

[130] Y. Raita, T. Goto, M. K. Faridi, D. F. Brown, C. A. Camargo, and K. Hasegawa, "Emergency department triage prediction of clinical outcomes using machine learning models," *Critical Care*, vol. 23, article no. 64, 2019.

[131] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.

[132] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[133] S. Z. Selim and K. Alsultan, "A simulated annealing algorithm for the clustering problem," *Pattern Recognition*, vol. 24, no. 10, pp. 1003–1008, 1991.

[134] W. Sheng, P. Shan, S. Chen, Y. Liu, and F. E. Alsaadi, "A niching evolutionary algorithm with adaptive negative correlation learning for neural network ensemble," *Neurocomputing*, vol. 247, pp. 173–182, 2017.

[135] Y. Shi and R. C. Eberhart, "Parameter selection in particle swarm optimization," In: *Proceedings of the 7th International Conference on Evolutionary Programming*, San Diego, USA, Mar. 1998, pp. 591–600.

[136] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," In: *Proceedings of the 1999 IEEE Congress on Evolutionary Computation*, Washington DC, USA, Jul. 1999, pp. 1945–1950.

[137] N. Siddique and H. Adeli, "Nature inspired computing: an overview and some future directions," *Cognitive Computation*, vol. 7, no. 6, pp. 706–714, 2015.

[138] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[139] Q. Song and Z. Wang, "Neural networks with discrete and distributed time-varying delays: a general stability analysis," *Chaos, Solitons and Fractals*, vol. 37, no. 5, pp. 1538–1547, 2008.

[140] B. Song, Z. Wang, and L. Zou, "On global smooth path planning for mobile robots using a novel multimodal delayed PSO algorithm," *Cognitive Computation*, vol. 9, no. 1, pp. 5–17, 2017.

[141] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," In: *Proceedings of the KDD Workshop on Text Mining*, Boston, USA, Aug. 2000, vol. 400, no. 1, pp. 525–526.

[142] P. N. Suganthan, "Particle swarm optimiser with neighbourhood operator," In: *Proceedings of 1999 IEEE Congress on Evolutionary Computation*, Washington DC, USA, Jul. 1999, pp. 1958-1962.

[143] C. Sui, M. Bennamoun, and R. Togneri, "Deep feature learning for dummies: a simple auto-encoder training method using particle swarm optimisation," *Pattern Recognition Letters*, vol. 94, pp. 75–80, 2017.

[144] W. Sun, A. Lin, H. Yu, Q. Liang, and G. Wu, "All-dimension neighborhood based particle swarm optimization with randomly selected neighbors", *Information Sciences*, vol. 405, pp. 141–156, 2017.

[145] M. Taherkhani and R. Safabakhsh, "A novel stability-based adaptive inertia weight for particle swarm optimization," *Applied Soft Computing*, vol. 38, pp. 281–295, 2016.

[146] P. N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, Boston: Addison Wesley, 2005.

[147] B. Tang, Z. Zhu, H. S. Shin, A. Tsourdos, and J. Luo, "A framework for multi-objective optimisation based on a new self-adaptive particle swarm optimisation algorithm", *Information Sciences*, vol. 420, pp. 364–385, 2017.

[148] Q. Tang, Y. Shen, C. Hu, J. Zeng, and W. Gong, "Swarm intelligence: based cooperation optimization of multi-modal functions," *Cognitive Computation*, vol. 5, no. 1, pp. 48–55, 2013.

[149] Y. Tang, Z. Wang, and J. Fang, "Parameters identification of unknown delayed genetic regulatory networks by a switching particle swarm optimization algorithm," *Expert Systems with Applications*, vol. 38, pp. 2523–2535, 2011.

[150] F. Tao, D. Zhao, Y. Hu, and Z. Zhou, "Resource service composition and its optimal-selection based on particle swarm optimization in manufacturing grid system", *IEEE Transactions on Industrial Informatics*, vol. 4, no. 4, pp. 315–327, 2008.

[151] R. Thangaraj, M. Pant, and A. Abraham, "A new diversity guided particle swarm optimization with mutation," In: *Proceedings of the 2009 World Congress on Nature & Biologically Inspired Computing*, Coimbatore, India, Dec. 2009, pp. 294-299.

[152] R. W. Tothill, A. V. Tinker, J. George, R. Brown, S. B. Fox, S. Lade, D. S. Johnson, M. K. Trivett, D. Etemadmoghadam, B. Locandro, and N. Traficante, "Novel molecular subtypes of serous and endometrioid ovarian cancer linked to clinical outcome," *Clinical Cancer Research*, vol. 14, no. 16, pp. 5198–5208, 2008.

[153] D. W. Van der Merwe and A. P. Engelbrecht, "Data clustering using particle swarm optimization," In: *Proceedings of the IEEE Congress on Evolutionary Computation*, Canberra, Australia, Dec. 2003, vol. 1, pp. 215–220.

[154] F. Van den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, 2004.

[155] J. Vissers and R. Beech, *Health Operations Management: Patient Flow Logistics in Health Care*, London: Routledge, 2005.

[156] L. Wan, M. Zeiler, S. Zhang, Y. Le Cun, and R. Fergus, "Regularization of neural networks using dropconnect," In: *Proceedings of the 30th International conference on machine learning*, Atlanta, GA, USA, Jun. 2013, pp. 1058–1066.

[157] X. Wan, Z. Wang, M. Wu, and X. Liu, "State estimation for discrete time-delayed genetic regulatory networks with stochastic noises under the Round-Robin protocols," *IEEE Transactions on Nanobioscience*, vol. 17, no. 2, pp. 145–154, 2018.

[158] F. Wang, H. Zhang, K. Li, Z. Lin, J. Yang, and X. L. Shen, "A hybrid particle swarm optimization algorithm using adaptive learning strategy", *Information Sciences*, vol. 436, pp. 162–177, 2018.

[159] H. Wang, Y. Jin, and J. Doherty, "Committee-based active learning for surrogate-assisted particle swarm optimization of expensive problems", *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2664–2677, 2017.

[160] 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.

[161] H. Wang, Y. Jin, and J. O. Jansen, "Data-driven surrogate-assisted multiobjective evolutionary optimization of a trauma system," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 6, pp. 939–952, 2016.

[162] L. Wang, Z. Wang, G. Wei, and F. E. Alsaadi, "Observer-based consensus control for discrete-time multi-agent systems with coding-decoding communication protocol," *IEEE Transactions on Cybernetics*, doi: 10.1109/TCYB.2018.2863664.

[163] L. Wang, Z. Wang, Q.-L. Han, and G. Wei, "Synchronization control for a class of discrete-time dynamical networks with packet dropouts: a coding-decoding-based approach," *IEEE Transactions on Cybernetics*, vol. 48, no. 8, pp. 2437–2448, 2018.

[164] Y. Wang, B. Li, T. Weise, J. Wang, B. Yuan, and Q. Tian, "Self-adaptive learning based particle swarm optimization", *Information Sciences*, vol. 181, no. 20, pp. 4515–4538, 2011.

[165] Z. Wang, Y. Liu, and X. Liu, "On global asymptotic stability of neural networks with discrete and distributed delays," *Physics Letters A*, vol. 345, no. 4, pp. 299–308, 2005.

[166] Z. Wang, L. Wang, S. Liu, and G. Wei, "Encoding-decoding-based control and filtering of networked systems: insights, developments and opportunities," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 1, pp. 3–18, 2018.

[167] L. Wang, Z. Wang, Q.-L. Han, and G. Wei, "Event-based variance-constrained $\mathcal{H}_\infty$ filtering for stochastic parameter systems over sensor networks with successive missing measurements," *IEEE Transactions on Cybernetics*, vol. 48, no. 3, pp. 1007–1017, 2018.

[168] L. Wang, Z. Wang, G. Wei, and F. E. Alsaadi, "Finite-time state estimation for recurrent delayed neural networks with component-based event-triggering protocol," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 4, pp. 1046–1057, 2018.

[169] P. N. Wright, G. Tan, S. Iliffe, and D. Lee, "The impact of a new emergency admission avoidance system for older people on length of stay and same-day discharges," *Age and Ageing*, vol. 43, no. 1, pp. 116–121, 2014.

[170] G. Wu, W. Lu, G. Gao, C. Zhao, and J. Liu, "Regional deep learning model for visual tracking," *Neurocomputing*, vol. 175, pp. 310–323, 2016.

[171] G. Wu, R. Mallipeddi, and P. N. Suganthan, "Ensemble strategies for population-based optimization algorithms – a survey," *Swarm and Evolutionary Computation*, vol. 44, pp. 695–711, 2019.

[172] G. Wu, X. Shen, H. Li, H. Chen, A. Lin, and P. N. Suganthan, "Ensemble of differential evolution variants", *Information Sciences*, vol. 423, pp. 172–186, 2018.

[173] X. Xiao, E. R. Dow, R. Eberhart, Z. B. Miled, and R. J. Oppelt, "Gene clustering using self-organizing maps and particle swarm optimization," In: *Proceedings of*

the International Parallel and Distributed Processing Symposium, Nice, France, April. 2003.

[174] B. Xue, M. Zhang, and W. N. Browne, "Particle swarm optimization for feature selection in classification: a multi-objective approach," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1656–1671, 2013.

[175] F. Yang, H. Dong, Z. Wang, W. Ren, and F. E. Alsaadi, "A new approach to non-fragile state estimation for continuous neural networks with time-delays," *Neurocomputing*, vol. 197, pp. 205–211, 2016.

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

[177] J. Yi, J. Bai, W. Zhou, H. He, and L. Yao, "Operating parameters optimization for the aluminum electrolysis process using an improved quantum-behaved particle swarm algorithm", *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3405–3415, 2018.

[178] S. Yu, Y. M. Wei, and K. Wang, "A PSO-GA optimal model to estimate primary energy demand of China," *Energy Policy*, vol. 42, pp. 329-340, Mar. 2012.

[179] D. Yu and L. Deng, "Deep learning and its applications to signal and information processing," *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 145–154, 2011.

[180] Y. Yu, H. Dong, Z. Wang, W. Ren, and F. E. Alsaadi, "Design of non-fragile state estimators for discrete time-delayed neural networks with parameter uncertainties," *Neurocomputing*, vol. 182, pp. 18–24, 2016.

[181] X. Yu, W.-N. Chen, T. Gu, H. Zhang, H. Yuan, S. Kwong and J. Zhang, "Set-based discrete particle swarm optimization based on decomposition for permutation-based multiobjective combinatorial optimization problems," *IEEE Transactions on Cybernetics*, vol. 48, no. 7, pp. 2139–2153, 2018.

[182] Y. Yuan, and F. Sun, "Delay-dependent stability criteria for time-varying delay neural networks in the delta domain," *Neurocomputing*, vol. 125, pp. 17–21, 2014.

[183] W. Yuan, Y. Liu, H. Wang and Y. Cao, "A geometric structure-based particle swarm optimization algorithm for multiobjective problems", *IEEE Transactions on Systems, Man, and Cybernetics–Part A: Systems*, vol. 47, no. 9, pp. 2516-2537, 2017.

[184] C. Yue, B. Qu, and J. Liang, "A multiobjective particle swarm optimizer using ring topology for solving multimodal multiobjective problems", *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 5, pp. 805–817, 2018.

[185] N. Zeng, Z. Wang, Y. Li, M. Du and X. Liu, "A hybrid EKF and switching PSO algorithm for joint state and parameter estimation of lateral flow immunoassay models," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 9, no. 2, pp. 321–329, 2012.

[186] N. Zeng, Y. S. Hung, Y. Li, M. Du, "A novel switching local evolutionary PSO for quantitative analysis of lateral flow immunoassay," *Expert Systems with Application*, vol. 41, no. 4, pp. 1708–1715, 2014.

[187] N. Zeng, H. Zhang, Y. Chen, B. Chen, and Y. Liu, "Path planning for intelligent robot based on switching local evolutionary PSO algorithm," *Assembly Automation*, vol. 36, no. 2, pp. 120–126, 2016

[188] N. Zeng, Z. Wang, H. Zhang, and F. E. Alsaadi, "A novel switching delayed PSO algorithm for estimating unknown parameters of lateral flow immunoassay," *Cognitive Computation*, vol. 8, no. 2, pp. 143–152, 2016.

[189] N. Zeng, Z. Wang, H. Zhang, W. Liu, and F. E. Alsaadi, "Deep belief networks for quantitative analysis of gold immunochromatographic strip," *Cognitive Computation*, vol. 8, no. 4, pp. 684–692, 2016.

[190] N. Zeng, H. Zhang, B. Song, W. Liu, Y. Li, and A. M. Dobaie, "Facial expression recognition via learning deep sparse autoencoders," *Neurocomputing*, vol. 273, pp. 643–649, 2018.

[191] N. Zeng, H. Qiu, Z. Wang, W. Liu, H. Zhang, and Y. Li, "A new switching-delayed-PSO-based optimized SVM algorithm for diagnosis of Alzheimer's disease," *Neurocomputing*, vol. 320, pp. 195-202, Dec. 2018.

[192] Z.-H. Zhan, J. Zhang, Y. Li, and H. S.-H. Chung, "Adaptive particle swarm optimization," *IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics*, vol. 39, no. 6, pp. 1362–1381, 2009.

[193] Z.-H. Zhan, J. Zhang, Y. Li, and Y.-H. Shi, "Orthogonal learning particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 6, pp. 832–847, 2011.

[194] Z.-H. Zhan, J. Li, J. Cao, J. Zhang, H. S.-H. Chung, and Y.-H. Shi, "Multiple populations for multiple objectives: a coevolutionary technique for solving multi-objective optimization problems," *IEEE Transactions on Cybernetics*, vol. 43, no. 2, pp. 445–463, 2013.

[195] C. Zhang, D. Ouyang, and J. Ning, "An artificial bee colony approach for clustering," *Expert Systems with Applications*, vol. 37, no. 7, pp. 4761–4767, 2010.

[196] H. Zhang, X. Cao, J. K. Ho, and T. W. Chow, "Object-level video advertising: an optimization framework," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 520–531, 2017.

[197] H. Zhang, Y. Peng, L. Hou, G. Tian, and Z. Li, "A hybrid multi-objective optimization approach for energy-absorbing structures in train collisions," *Information Sciences*, vol. 481, pp. 491–506, 2019.

[198] J. Zhang, L. Ma, and Y. Liu, "Passivity analysis for discrete-time neural networks with mixed time-delays and randomly occurring quantization effects," *Neurocomputing*, vol. 216, pp. 657–665, 2016.

[199] K. Zhang, Q. Huang, and Y. Zhang, "Enhancing comprehensive learning particle swarm optimization with local optima topology", *Information Sciences*, vol. 471, pp. 1–18, 2019.

[200] S. Zhang, J. Xu, L. H. Lee, E. P. Chew, W. P. Wong, and C. H. Chen, "Optimal computing budget allocation for particle swarm optimization in stochastic optimization", *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 2, pp. 206–219, 2017.

[201] X. Zhang, X. Wang, Q. Kang, and J. Cheng, "Differential mutation and novel social learning particle swarm optimization algorithm," *Information Sciences*, vol. 480, pp. 109–129, 2019.

[202] X. Zhang, X. Zheng, R. Cheng, J. Qiu, and Y. Jin, "A competitive mechanism based multi-objective particle swarm optimizer with fast convergence", *Information Sciences*, vol. 427, pp. 63–76, 2018.

[203] Y. F. Zhang, and H. D. Chiang, "A novel consensus-based particle swarm optimization-assisted trust-tech methodology for large-scale global optimization", *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2717–2729, 2017.

[204] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, "Deep learning and its applications to machine health monitoring," *Mechanical Systems and Signal Processing*, vol. 115, pp. 213–237, 2019.

[205] Q. Zhu, Q. Lin, W. Chen, K. C. Wong, C. A. C. Coello, J. Li, J. Chen, and J. Zhang, "An external archive-guided multiobjective particle swarm optimization algorithm", *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2794–2808, 2017.