Novel Particle Swarm Optimization Algorithms with Applications in Power Systems

Izaz Ur Rahman

College of Engineering, Design & Physical Sciences

Brunel University London, United Kingdom

A thesis submitted for the degree of

Doctor of Philosophy

September 2015

Dedicated to

The sweet memories of my late father. I wish he was alive today and I could sit at his feet to express my feelings and emotions to him. To my mother for her tremendous support, encouragements, motivations and her sincere prayers for my success. To my daughter Sara and my son Ayyan for their tolerance during my studies. They were in the age, where they needed me a lot. To my wife for her love, patience and support during the long period of my studies. Last but not least, I dedicate this thesis to my supervisors Professor Zidong Wang and Professor Xiaohui Liu.

Acknowledgements

I am extremely obliged to my supervisors Professor Zidong Wang and Professor Xiaohui Liu for their whole-hearted guidance and instructions during my research. They have made my experience of Ph.D much more attractive, productive and innovative. Thank you so much for all your support, encouragement and prompt response. In the rainy days, I always found myself under the umbrella of their support. I feel blessed and honoured to work for such a long period with them. I am grateful to many of my colleagues who supported me in the hard time especially Liang Hu, Dr.Mukhtaj Khan, Muhammad Suleman Riaz, Krishnanand, Dr.Shariq Mahmood Khan, Rahim Khan and Muhammad Zakarya. I would like to acknowledge the help of all my colleagues in the CIDA group. I am grateful to all my siblings, particularly my big brother Arshur Rahman for his support. In fact, he was impatient about my Ph.D more than me. Last but not least, especial thanks to my sponsor Abdul Wali Khan University Mardan, Pakistan, for providing full financial support throughout my studies. Indeed, it was impossible without their support.

Abstract

Optimization problems are vital in physical sciences, commercial and finance matters. In a nutshell, almost everyone is the stake-holder in certain optimization problems aiming at minimizing the cost of production and losses of system, and also maximizing the profit. In control systems, the optimal configuration problems are essential that have been solved by various newly developed methods. The literature is exhaustively explored for an appropriate optimization method to solve such kind of problems.

Particle Swarm Optimization is found to be one of the best among several optimization methods by analysing the experimental results. Two novel PSO variants are introduced in this thesis. The first one is named as N State Markov Jumping Particle Swarm Optimization, which is based on the stochastic technique and Markov chain in updating the particle velocity. We have named the second variant as N State Switching Particle Swarm Optimization, which is based on the evolutionary factor information for updating the velocity. The proposed algorithms are then applied to some widely used mathematical benchmark functions. The statistical results of 30 independent trails illustrate the robustness and accuracy of the proposed algorithms for most of the benchmark functions. The better results in terms of mean minimum evaluation errors and the shortest computation time are illustrated.

In order to verify the satisfactory performance and robustness of the proposed algorithms, we have further formulated some basic applications in power system operations. The first application is about the static Economic Load Dispatch

and the second application is on the Dynamic Economic Load Dispatch. These are highly complex and non-linear problems of power system operations consisting of various systems and generator constraints. Basically, in the static Economic Load Dispatch, a single load is considered for calculating the cost function. In contrast, the Dynamic Economic Load Dispatch changes the load demand for the cost function dynamically with time. In such a challenging and complex environment the proposed algorithms can be applied. The empirical results obtained by applying both of the proposed methods have substantiated their adaptability and robustness into the real-world environment. It is shown in the numerical results that the proposed algorithms are robust and accurate as compared to the other algorithms. The proposed algorithms have produced consistent best values for their objectives, where satisfying all constraints with zero penalty.

Contents

Abbreviations

AI	**A**rtificial **I**ntelligence
ABC	**A**nt **B**ee **C**olony
ACO	**A**nt **C**olony **O**ptimisation
ANN	**A**rtificial **N**eural **N**etwork
APSO	**A**daptive **P**article **S**warm **O**ptimisation
Bb-PSO	**B**arebones **P**article **S**warm **O**ptimisation
CPSO	**C**o-operative **P**article **S**warm **O**ptimisation
CLPSO	**C**ompetitive **L**earning **P**article **S**warm **O**ptimisation
DE	**D**ifferential **E**volution
DEED	**D**ynamic **E**conomic **E**mission **D**ispatch
DELD	**D**ynamic**E**conomic **L**oad **D**ispatch
DMS-PSO	**D**ynamic **M**ulti **S**warm **P**article **S**warm **O**ptimisation
EA	**E**volutionary **A**lgorithm
EA-PSO	**E**nhanced **A**daptive-**P**article **S**warm **O**ptimisation
EC	**E**volutionary**C**omputation
EF	**E**volutionary **F**actor
EKF	**E**xtended **K**alman **F**ilter
ELD	**E**conomic **L**oad **D**ispatch
EP	**E**volutionary **P**rogramming
ES	**E**volutionary **S**trategies
ESE	**E**volutionary **S**tate **E**stimation
FE	**F**unction **E**valuations
FESPSO	**F**itness **E**volution **S**trategy **P**article **S**warm **O**ptimisation
FIPS	**F**ully **I**nformed **P**article **S**warm
FSL-PSO	**F**uzzy **S**elf-adaptive **L**earning-**P**article **S**warm **O**ptimisation
GA	**G**enetic **A**lgorithm
Gbest	**G**lobal **B**est
GP	**G**enetic **P**rogramming
GPD	**G**aussian **P**robability **D**istribution
GPSO	**G**lobal-version **P**article **S**warm **O**ptimisation
GRN	**G**enetic **R**egulatory **N**etworks
HC	**H**ill **C**limbing
HPSO	**H**ybrid **P**article **S**warm **O**ptimisation
HS	**H**armony **S**earch
IPSO	**I**ntelligent **P**article **S**warm **O**ptimisation

LCPMA	**L**eader **C**ompetitive **P**enalized **M**ulti-learning **A**pproach
LPSO	**L**ocal-version **P**article **S**warm **O**ptimisation
MJ	**M**arkovian **J**umping
MPSO	**M**odified **P**article **S**warm **O**ptimisation
NS-MJPSO	**N** **S**tate **M**arkovian **J**umping **P**article **S**warm **O**ptimisation
NS-SPSO	**N** **S**tate **S**witching **P**article **S**warm **O**ptimisation
NPSO	**N**ew **P**article **S**warm **O**ptimisation
NPSO-LRS	**N**ew **P**article **S**warm **O**ptimisation-**L**ocal **R**andom **S**earch
Pbest	**P**ersonal **B**est
PSO	**P**article **S**warm **O**ptimisation
PSO-LDIW	**P**article **S**warm **O**ptimisation- **L**inearly **D**ecreasing **I**ertia **W**eight
PSO-CF	**P**article **S**warm **O**ptimisation-**C**onstriction **F**actor
PSO-TVAC	**P**article **S**warm **O**ptimisation **T**ime **V**arying **A**cceleration **C**oefficients
RS	**R**andom **S**earch
SA	**S**imulated **A**nnealing
SL-PSO	**S**ocial **L**earning **P**article **S**warm **O**ptimisation
SPSO	**S**witching **P**article **S**warm **O**ptimisation
TS	**T**abu **S**earch
TVAC-IPSO	**T**ime **V**arying **A**cceleration **C**oefficients-**I**mproved **PSO**
UC	**U**nit **C**ommitment
WNN	**W**avelet **N**eural **N**etwork

Author's Publications and Presentations

Journal: Liang Hu, Zidong Wang, **Izaz Rahman** and Xiaohui Liu. "A Constrained Optimization Approach to Dynamic State Estimation for Power Systems Including PMU and Missing Measurements." Accepted in Control Systems Technology, IEEE Transactions

Conference: Zidong Wang, Liang Hu, **Izaz Rahman** and Xiaohui Liu. "A Constrained Optimization Approach to Dynamic State Estimation for Power Systems Including PMU Measurements." In Automation and Computing (ICAC), Proceedings of the 19th International Conference on, pp. 1-6. IEEE, 2013.

Chapter 1

Introduction

1.1 Motivation of the Research

The importance of research has been revealed fifteen hundred years ago referring to the explanation of Quranic verse as follows:

"Observe! In the creation of the heavens and the earth; in the alternation of the night and the day; in the sailing of the ships through the ocean for the benefit of mankind; in the rain which Allah sends down from the skies, and the life which He gives therewith to an earth that is dead; in the beasts of all kinds that He scatters through the earth; in the change of the winds, and the clouds which they trail like their slaves between the sky and the earth – (Here) indeed are signs for people that are wise. (Surah Al-Baqarah, 2:164)"

To get the ball rolling, we present the topic of optimization. In general term, optimization is concerned with the life of everyone in the real-world, which is an individual problem or a community one. The basic idea is about minimization or maximization of an objective according to the constraints applied to the problem. Furthermore, a particular solver is selected by studying the nature of the problem. First of all, the problem has to be categorized into a specific class. The categorization is further based on mathematical and analytical grounds. The main categories are commonly known as linear and non-linear, static and dynamic, convex and non-convex, smooth and non-smooth. These terms are sometimes used interchangeably, however, each one has its own meaning and background. The problems that have linearity, differentiability and smoothness in the objective functions are relatively simple and easy to solve. In contrast, non-linear, non-convex and non-differentiable functions are much more challenging. Subsequently, the solutions are categorized into two cases. The first case is the classical or conventional methods commonly used for linear type and differentiable problems. The use of classical methods is discouraged by conditional entailment to the problem. The second case is about heuristic methods. These methods imitate the process of learning behaviour of animals such as birds, fish, bees and other natural phenomenon. These methods are very effective, when used to solve many real-world problems. The solution is not guaranteed to be a true optimum, however, an approximate or near optimum solution is always produced. Moreover, stochastic or randomized terms have been introduced and explored in detail with the applications in the real-world problems. An optimum is predicted with a certain probability by randomly searching in the boundary space. Genetic Algorithm (GA), Evolutionary Computation (EC), Simulated Annealing (SA), Tabu Search (TS), Hill Climbing (HC), Ant Bee Colony (ABC) and Particle Swarm Optimization (PSO) algorithms have been applied to solve real-world problems Chapter 2. Furthermore, all the above heuristic methods have their strengths and weaknesses. The one which is most commonly used is GA, which has

its own characteristics to find the optimal solution for complex problems. However, there are some limitations concerned with GA, that are also common in other heuristic methods. The GA has the complicated implementation procedure, where the best optimum value is dependant on the initial population or initial state. An excessive number of evaluations are required for a specific problem and sometimes the algorithm takes days to compile results. These limitations makes GA inefficient and unattractive.

Particle Swarm Optimisation (PSO) is a population based technique first developed in 1995 by Kennedy and Eberhart [36; 37]. The idea is taken from fish schooling, birds flocking, and the learning behaviour of each individual in the swarm. The population of fish, birds, or agents is called swarm. It has further generalized the area for research as swarm intelligence. Each member of the swarm is a candidate solution, which is named as particle here. The main characteristics of PSO algorithm are its simple structure, easy implementation and quick convergence. PSO algorithm has widely been used by many researchers for various engineering optimization problems. It has also been used for artificial neural networks training. PSO has been developed with the intention to address the limitations of the existing methods, particularly GA. PSO algorithm has the simple structure and easy implementation procedure. Two main terms velocity and position are used which controls the flow of entire algorithm with the help of cognitive, social and inertia weight parameters. The simple structure, easy implementation and quick convergence have enchanted the greater part of the research community. It has been investigated by applying to numerous real-world problems. The strengths and limitations of PSO algorithms have been highlighted as the motivations for further improvements. The basic PSO algorithm has been gradually modified and improved in many aspects. Some other additional stochastic and mathematical methods have been involved to obtain the maximum performance of PSO algorithm. However, each variant is designed with a specific mechanism for an individual problem. It may produce better results for a specific problem with current parameters settings. However, some adjustment will be required to fit into other problems. In order to investigate the strength of modified PSO algorithms, complex and dynamic problems in the real-world have also to be evaluated. The power system is one of the world's most important, huge, complex, and expensive systems. In our daily life, everyone needs electricity and the main problem is the exponentially increasing demand for electricity production. Extraordinary funds and time are needed to expand the capacity of power generation. In this case, to obtain the maximum performance of the existing power generation system in a reliable and affordable way is the primary task. Optimization is required in every part of the power system. In this thesis, we consider the static ELD and Dynamic Economic

Load Dispatch, which is part of the Unit Commitment problem.

In summary, the main emphasis of this thesis is the development of novel PSO algorithms and later solve the complex and dynamic problems of power system operations. The applications are used to test the diversity of newly developed algorithms in this complex environment.

1.2 Goals of the Thesis

1.2.1 The state-of-the-art for Particle Swarm Optimization

In search of a suitable method for solving complex problems, meta-heuristics have been widely explored and several algorithms have been developed in the last few decades. Amongst all of algorithms, this study is focused on the state-of-the-art for Particle Swarm Optimization (PSO) algorithm. As we have discussed briefly about the nature of complex problems, dynamic optimization is a challenging task. Apart from the problem, PSO algorithm has the capability of combining and merging other techniques into it. Modification for improvement in performance is always welcomed by researchers. In the last two decades, many variants have been developed and many applications have been made. Due to its simplicity, the PSO algorithm can be easily modified and implemented to many different kinds of optimization problem.

The PSO algorithm has few parameters that are cognitive, social and inertia weights. Each parameter has strong influence to control the movement of particle. The adjustment of parameter is also a challenging task. Therefore, the recent research is focused on the automatic parameter adaptation according to the dimensionality and nature of the problem. Basically, as a member of heuristics techniques family, PSO learns through the social and personal learning behaviours. The second topic that has recently been considered for improvement in the existing PSO algorithm is the topological improvement. It is our basic concern to reduce computation complexity, computation time and also to simplify implementation procedure. In our newly developed algorithms, we aim for satisfying the aforementioned objectives.

In this section, some publications are considered to be the bases for research proposed and developed in this thesis. In the following, we will present a serial connection of this thesis to the existing methods.

The enhancement in speed of global convergence, and avoiding premature local convergence are the main objectives to be achieved in the newly developed algorithms. Several works have been conducted to achieve these objectives [1; 2; 3; 4]. In this development,

adjusting the parameters of the algorithm, the association of auxiliary search operators and augmenting topological structure have turned into the three most conspicuous approaches [5]. Conversely, still, it has found to be challenging to achieve both the goals simultaneously. The comprehensive-learning PSO (CLPSO) [4] emphasises on the escaping from local optima, but results in a slower convergence.

In the successive revisions of PSO algorithm, the prime objective that has been considered is to minimize the complexity in the algorithm. The problem of local optimum or premature convergence has also been examined thoroughly in [1; 2; 3; 4]. The improved algorithms have been further verified by doing some applications in real-world environment. Due to the indeterminate, non-linear and complex nature of the real-world problems, there is always a gap for enhancement in the current state algorithms. Apparent to that, the auxiliary skills have been applied to control the parameters significantly [6; 7; 8]. The topological structures have been developed to achieve the second objective to ensure global optimum and avoid premature convergence [4].

Adaptive PSO (APSO) [7] has concentrated on the global convergence and topological enhancement. The evolutionary factor E_f has further been added the population distribution characteristics via the mean distance between the global best and other particles in the swarm. Four states, S_1, S_2, S_3, and S_4, have been described by taking population distribution information E_f into account, which indicates convergence, exploration, exploitation and jumping out states respectively. Fuzzy classification method has been used which has resulted some limitations of 1). excessive computation of acceleration coefficients in each generation, 2). swarm stagnation in the local optima if the current global best is the local optimum, and 3). is the complicated implementation of classification method.

Switching PSO (SPSO) in [8] has been developed to further improve the search performance of APSO algorithm and addressed all its limitations. A novel approach, the Markovian switching has been used to avoid getting trapped in to local optima or premature convergence, and leader competitive penalized multi-learning approach (LCPMA) has been used to improve global search performance [9]. Same four states as described in [7] have been used here. However, a new mode-dependent technique with Markovian Jumping parameters has been adopted in the velocity update equation to improve the search performance. SPSO has produced promising results in [8; 9]. SPSO has been re-evaluated in comparison with some other state-of-the-art algorithms on some benchmark problems. It has been observed that SPSO is very problem specific, and it requires much parameter adjustment to perform better in the dynamic environment or any other new problem. However, to enhance the performance of SPSO, it has been further modified in this thesis

by extending the concept of four states into N states.

1.2.2 Main Contributions in the Thesis

Several methods have been proposed to solve optimization problems. In the comparative analysis some methods have performed better. Basically, to check the effectiveness of a specific method, significant computation capacity is required to complete a task in a complex environment. Actually, by complex environment we refer to the real-world problem. However, steady growth have been observed in the development of a competent optimization method. As far the selection of the real-world applications are concerned, we have chosen static Economic Load Dispatch (ELD) and Dynamic Economic Load Dispatch (DELD). These are the power systems operations problems of highly non-linear and complex nature. Both of the problems have non-linear and non-smooth constraints with their objectives. Basically, several algorithms have been already applied for the same type of problems, but a robust and more accurate algorithm is a challenging task. Therefore, the main aim of this thesis is to explore, examine and develop novel algorithms capable of solving a particular real-world problem efficiently with more accurate and robust performance.

To show how we achieve the above mentioned aims and objectives, we summarise the main contributions of this thesis as follows.

(I) The first contribution of this thesis is the development of a new algorithm named as N State Markov Jumping Particle Swarm Optimization (NS-MJPSO). The proposed algorithm is developed in the inspiration of a serial connection of algorithms, that are APSO [7] and SPSO [8]. The further improvement of these algorithms and extending their capability is the main concern in this thesis. Evolutionary concept, population distribution, stochastic technique and Markov chain have been used. A new mechanism of N states is proposed here. The better understanding of the problem nature is required to determine the number of states accurately. Where N can take any value. Markov chain is introduced here to keep the particle moving in the search space. Therefore, the problem of premature convergence is solved by the proposed algorithm. The N number of acceleration coefficients are pre-defined and then the appropriate coefficient value is assigned in each of the current states. The initial transition probability $\Phi = 0.9$ is set for the randomly chosen current state, and then the next state is predicted. A mode-dependent switching parameter is introduced in the velocity update equation of particle swarm optimization. The newly developed

NS-MJPSO is then applied to some benchmark functions for validating the performance. The simulation results have concluded the best overall performance of the proposed algorithm.

(II) The second contribution of this thesis is the development of another new algorithm named N State Switching Particle Swarm Optimization (NS-SPSO). The main idea of the NS-SPSO is similar to the algorithm NS-MJPSO. In addition, with all the advantages of stochastic NS-MJPSO algorithm, we have also observed some limitations in the analysis of parameters experimental settings. For the use of NS-MJPSO, we need some background knowledge of the problem in hand for optimization. For example, how to set the jump probability and number of states N. The second problem is the increasing computation time and complexity in NS-MJPSO. We can apply NS-MJPSO with increased states when the main interest is the accuracy of the result and the computation time is not very important. Therefore, to address these limitations, in the new algorithm we exclude the stochastic term Markov chain from the body of the algorithm. Yet the N states concept is still used. Time varying concept is introduced for the measurement of inertia weight ω. NS-SPSO is based on evolutionary technique. The resultant algorithm is applied to the same benchmark functions used for NS-MJPSO algorithm. The results are then compared with various state-of-the-art PSO variants including NS-MJPSO algorithm. It has been observed that the novel NS-SPSO has performed very well in comparison.

(III) As the third contribution, we have applied both of the algorithms, NS-MJPSO and NS-SPSO to solve Static Economic Load Dispatch (ELD) problem. The main reason and inspiration for applying new algorithms is the enormous application development of basic PSO and its variants for ELD problems [10; 11; 12; 13]. Basically, ELD is a complex, highly non-linear and non-smooth function. The main objective is to minimize the total generation cost while meeting the load demand and satisfy the constraints. Another recently published SL-PSO [14] has also been applied to the same problem. The performance of all three algorithms is then evaluated by solving several case studies having 6, 13, 15, 40, and 140 unit systems, which represents from small and simple 6 to large and complicated 140 unit systems. Each case study has its own dimensionality, maximum minimum capacity limits and other related data settings. Promising results have been produced by all the three algorithms.

(IV) The fourth main contribution describes the application of the newly developed NS-MJPSO and NS-SPSO algorithm in Dynamic Economic Load Dispatch (DELD)

problems [15; 16; 17; 18]. The static ELD refers to the single load within the entire compilation of the algorithm. Whereas, the DELD needs to change the load with time. 24 hours dispatch is considered for this problem. Therefore, we have 24 different loads to be optimally and economically scheduled with the defined limits and constraints of the system. Two case studies of 5, and 10 Unit system are selected. The results of the algorithms shows that the proposed algorithms have produced promising values for its objective with fast convergence.

1.3 Overview of the Chapters

The main work of thesis is organized chapter-wise in the following order:

Chapter 2: This chapter describes the exhaustive survey of the background literature related to evolutionary computation, genetic algorithms, particularly the developmental studies of PSO algorithm and its applications in many fields including power systems. The main subject for literature is the categorized survey of PSO algorithm development and their applications particularly in ELD and DELD problems in power system.

Chapter 3: This chapter provides the first method, the N State Markov Jumping Particle Swarm Optimization (NS-MJPSO) algorithm. This algorithm is based on stochastic technique of Markov chain and evolutionary factors. The method is exhaustively evaluated for various uni-modal and multi-modal benchmark functions and the results are compared with the existing well-known PSO variants.

Chapter 4: This chapter presents the second method, the N State Switching Particle Swarm Optimization (NS-SPSO) algorithm. This algorithm is based on the evolutionary factors. The method is exhaustively evaluated for various uni-modal and multi-modal benchmark functions and the results are compared with the NS-MJPSO and other existing well-known PSO variants.

Chapter 5: This chapter introduces the first successful application of the newly developed two algorithms NS-MJPSO and NS-SPSO, and also the recently published SL-PSO [14]. The problem for application is the static Economic Load Dispatch (ELD). With aim to minimize the total generation cost while satisfying all constraints.

Chapter 6: This chapter introduces the second successful application of the newly developed two algorithms NS-MJPSO and NS-SPSO. The problem for application is the Dynamic Economic Load Dispatch (DELD). With aim to minimize the total generation cost with the dynamically changing load demands while satisfying all constraints.

Chapter 7: In this chapter, we conclude our thesis with the scope, limitations and the possible future work.

Chapter 2

Background

2.1 Introduction

Optimization is the real world problem, which can be defined as the minimization or maximization of a function and satisfying some constraints on that function. To solve such kind of problem, we model the real world problem mathematically. It has been observed that some problems are linear, while some are non-linear by nature. Basically, three different approaches have been adopted to solve certain optimization problems[19]. The first one is the deterministic approach which is based on the given initial condition or initial assumption. The second approach is the analytic approach which is also defined as the conventional approach of optimization. The analytical methods have the targets pre-defined for a specific problem. These types of methods might fail to work when the dimensionality of the problem increases dramatically. The third one is the stochastic approach which can always find the near optimum solution. We further elaborate stochastic approaches in the next section.

2.2 Stochastic Algorithms

The deterministic and analytical methods fail to find the optimal solution for high dimensional problems. The stochastic approaches are adopted for such kind of problems [20]. These approaches find the approximate solution, or nearly optimum solution, which means the global optimum is not guaranteed here. Stochastic approaches are very powerful in nature comparatively. These methods are easy to implement and appropriate for combinatorial problems because these methods do not need the function to be differentiable and continuous. Stochastic methods have the random probability distribution and these methods are analysed statistically without guarantee to be precise. Various stochastic algorithms have been developed for solving real world optimization problems, including Simulated Annealing, Random Search, Hill Climbing, Tabu Search, and Evolutionary Algorithms. These algorithms follow the same pattern for solving the problem iteratively. These algorithms have the following steps:

1. The current solution or initial solution is randomly created.
2. The current solution is further modified to find new solution.
3. The new solution is then evaluated and compared with the current solution. If the new solution is better than the current solution, we change the current solution to the new solution; otherwise we keep the same value for the current solution.

4. Repeat step 2 until the stopping criteria is met.

These methods require several iterations and perturbations to the initial random solution, which causes slow convergence, local optimum or premature convergence. To consider these problems we further explore evolutionary algorithms.

2.3 Evolutionary Algorithms

In evolutionary algorithms we generalise the area as evolutionary computation. The evolution is considered as a foundation term. Evolutionary algorithms are all based on population of candidate solutions, where the candidate solutions are randomly generated in a finite search space. All the candidate solutions are evolved to find their possible optimal solutions. There are several strategies and methods named as evolutionary process, evolutionary strategies, evolutionary programming etc. These are all different methods, having different processes. However, all of them are originated under the general term of evolutionary computation. The initialization of the population is the first step of each evolutionary computation algorithm. Numerous structures have been adopted to generate and initialize population randomly in the search space. The next step is formulating an objective or fitness evaluation function. This function is used for the evaluation of the entire population throughout the search space. Decisions are made on the basis of the derived function evaluation values. Some candidate solutions are selected as parents to produce new candidate solutions, called offspring or child. The process of evolution is described as two different techniques, which are crossover and mutation. In crossover technique the bits of parent candidates are twisted, exchanged, and recombined with each other to produce an offspring. Mutation changes the bits of one parent and thus produces a new offspring. The new population is generated through crossover and mutation. This new population of offspring is then evaluated on the given evaluation function or fitness function until it reaches its global optimum or maximum number of iterations.

Evolutionary algorithms have several reasons for being a suitable optimization method. For instance, problem domain knowledge is represented to generate population within its boundaries and dimensionality. We can also merge a conventional approach to evolutionary algorithms to make it a robust optimization method. This can absorb the dynamic variations in the problem. The main advantage of evolutionary algorithms is the adaptation of non-differentiable objective functions. The evolutionary algorithms were first proposed in 1950s and were evolved slowly because of the absence of high performance computing. However, the invention of less expensive and high speed processing power of computers has

diverted researchers attention to evolutionary algorithms. The evolutionary algorithms are further classified into four different techniques.

2.3.1 Evolutionary Programming

Fogel has developed evolutionary programming (EP) in 1966 [21; 22]. EP is used to find the optimum of real valued functions. The mutation operator is first introduced. Hence, the two basic operators Selection and Mutation are used. An individual candidate solution is composed of a set of real valued trajectories. The trajectories of the candidate solution represent the decision variables in the objective function and the parameters for mutation. Tournament selection is adopted in EP. The best candidate solution is always maintained for further iterations. Mutation is used by taking the uniform probability distribution. The rate of mutation decreases as it comes close to the optimum. One offspring is produced by one parent in every iteration. The N best offspring having the best fitness values are selected for the next generation. EP has also been applied in power systems [23; 24].

2.3.2 Evolutionary Strategies

Rechenberg in 1970's first introduced evolutionary strategies (ES) [25]. The ES adopts problem-dependent representations, and this method also uses mutation and selection for its search operations. ES is applied to real-valued search problems. There are two pairs of the real valued vectors. One trajectory consists of problem parameters that need to be considered for optimization, and the second consists of strategy parameters that are used for the mutation control. Two strategies, comma and plus are used for the next generation. Comma strategy describes, a portion of the offspring is presented to act as a parent in the next generation. In this strategy some best candidate solutions are wasted. The second strategy which is named as plus strategy operates the competition of parents and offspring to be preserved in the next generation. Various researchers introduced some modified and improved strategies regarding to ES, which develop the mutation operator by applying normal distributed random numbers to the set of its parameters. Several applications of ES have been presented in [26; 27; 28].

2.3.3 Genetic Algorithm

Bremermann and John Holland are known as the pioneers of Genetic Algorithms (GA). The idea is briefly described in [29; 30]. A lot of works have been done, and many variants have been presented and published for further improvements in operations of GA, such as structure for representation, creation of population, the selection criteria for a specific population, the process of mutation, and the process of crossover. Each individual of the population is known as chromosome and each chromosome is made of small genes. Each chromosome is represented as a string and each bit in that string is a gene. Initial population is generated randomly depending upon the nature of the problem. The crossover is applied as single point, multipoint, and uniform with more than 95% rate. The mutation is also applied with the least rate of 5% for avoiding stagnation and premature convergence. The elitism mechanism is also used survival of the fittest. The best fit individual is selected to take part in the next generation. GA has widely been applied in many areas and it has shown good results because of its evolutionary, flexible and adaptive nature [31; 32]. It has also been applied in power systems [33] and signal processing [34; 35].

2.3.4 Genetic Programming

Genetic programming is a special case of genetic algorithm or part of it. Genetic Programming uses computer programs as population. Each individual in the population is a computer program. The population of programs is modelled as parse trees. GP solves the problems in the following steps.

1. GP generates initial population randomly in different ways. Each individual is the structure of functions and terminals or nodes that are computer programs.
2. Each individual program is compiled and evaluated, a fitness function is assigned according to the problem domain because the fitness function is also problem specific.
3. Offspring are created in three different ways, which are called new population.
   (a) The individuals with best fitness values are copied as offspring.
   (b) New offspring are created from computer program by genetic mutation operator.
   (c) The genetic reproduction operator crossover is applied to create new population.
4. The current best individual or computer program which is selected in all generation is concluded as the result of GP.

2.4 Particle Swarm Optimization

PSO is a population based technique first developed in 1995 by Kennedy and Eberhart [36; 37]. The idea is taken from fish schooling, birds flocking, and the learning behaviour of each individual in the swarm. The population of fish, birds, or agents is called swarm. It has further generalized the area for research as swarm intelligence. Each member of the swarm is a candidate solution, which is named as particle here. The main characteristics of PSO algorithm are its simple structure, easy implementation and quick convergence. PSO algorithm has widely been used by many researchers for various engineering optimization problems. It has also been used for artificial neural networks training.

2.4.1 Traditional PSO Algorithm and its Framework

PSO is one of the most powerful and modern, population based, swarm intelligence technique for solving global optimization problems. The particles emulate the behaviour of fish school and bird flocks to find the objective of the real-world optimization problem. Population is referred as swarm, and each individual agent is the candidate solution referred to the term particle. The particles are represented as real valued according to the original basic PSO. Basically, two approaches are used for initializing the swarm in the beginning. The first one is the case where population is initialized randomly. The second approach is to initialize the swarm according the variables of the problem in consideration. Each particle i in the swarm has some attributes. Each particle is associated with two vectors, i.e. the velocity vector $v_i = [v_i^1, v_i^2, \cdots, v_i^D]$ and the position vector $x_i = [x_i^1, x_i^2, \cdots, x_i^D]$, where D is the dimensionality of the problem. The velocity and position are initialized randomly within the range of its search space. Throughout the development process, the velocity and position of ith particle are updated on dimension D according to the following two equations.

$$v_i(t+1) = v_i(t) + c_1 r_1(pbest_i(t) - x_i(t)) + c_2 r_2(gbest_i(t) - x_i(t)) \tag{2.1}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{2.2}$$

c_1 and c_2 are the acceleration coefficient weights normally having the constant value 2.0. r_1 and r_2 are two uniformly distributed random numbers belonging to $(0, 1)$. The velocity update equation has three terms on the right-hand side in which the first term is called momentum or inertia weight. The second term is called cognitive part which denotes the

personal influence of the particle. The third term is called social part which denotes the social and collective influence of the particle. The main role of the acceleration coefficients is to adjust the balance between exploitation and exploration of each particle in the search space during its movement. V_{max} is also introduced to further constrain the movement of particle within the boundary of the search space. The value of V_{max} is given according to the problem minimum and maximum boundary. The small V_{max} causes exploitation, and big V_{max} causes exploration. Each particle memorizes best positions it has found in the history. The best position that is found by each particle itself is called personal best P_{best} position. However, the best position found by the whole swarm is called global best or G_{best}. In the beginning, each particle moves with random velocity in the search space. Subsequently, each particle dynamically fine-tunes its movement velocity matching to the experiences of its own and other participants. The particle position will be updated constantly until the final optimal solution is found or maximum iteration is reached. Fig.2.1 describes the main process of traditional PSO algorithm.

Comparatively, PSO is one of the simplest and fastest method to find the best value and assure convergence stability. In contrast to the other algorithms, PSO has the flexible and balanced approach to adjust its movement in local and global search spaces. Though, having all the above mentioned qualities, PSO also has various shortcomings, it looks very sensitive to properly tune the parameters and weights for a specific problem. Further, the lack of diversity in the population some time cause stagnation, premature convergence or local convergence. Consequently, PSO has been a broad burning topic for researchers. To improve the poor search performance of the PSO algorithm, many variants have been developed. All the variants have been classified into five different categories [5]. In contribution to the context of this research, the development of an N State Markovian Jumping PSO is one of the aims of this work. In the following subsection we have explored the classification of PSO research directions.

2.4.2 PSO Research Directions

A sum of the latest studies has been explored for the development of a suitable PSO algorithm that can be applied into the complex environment. Most of the modifications have been made in PSO algorithm are based on applications. That's the quality of PSO algorithm to fit to the new environment, after a minor adjustment in its parameters. To generalize PSO algorithm for further research, numerous aspects have been considered for the improvement of traditional PSO. Cheng [14], have also designed the study into five main categories, which is listed as follows:

1. **Parameters Adaptation:** As we have mentioned in the previous paragraph that PSO has been applied in several areas, and it has been modified according to the nature and requirement of the problem. The main structure of the algorithm is maintained, however, some additional parameters have been introduced. In the following subsection we have discussed the primary parameters and some additional parameters of PSO algorithm.

   (a) **Maximum Velocity:** V_{max} was originally assigned a fixed value. Though, Fan, Shi [38; 39] introduced a dynamic strategy for V_{max}. Further, Abido [40; 41] has proposed the following dynamic strategy for V_{max}.

   $$V_{max} = \frac{X_{max} - X_{min}}{N} \tag{2.3}$$

   where X_{max} and X_{min} represent the boundaries values of particles in the swarm, while N is the maximum number of iterations.

   (b) **Acceleration Coefficients:** The movement of the particle in the search space, to the global optimum, is controlled by acceleration coefficients. Small values of c_1, the cognitive factor, causes local convergence, while large values causes exploration of the particle. c_2 is the social influence factor. The variation of its values effect the global convergence of the algorithm, this has been reported in [42]. PSO with time varying acceleration coefficients has been proposed by Ratnaweera [43], the algorithm is aimed to dynamically change the values of acceleration coefficients in each iteration. The improved behaviour of the swarm is observed throughout the course of maximum iteration. However, in most of the PSO applications a fixed value 2.0 is assigned to both of the acceleration coefficients.

   (c) **Inertia Weight:** The inertia weight factor denoted as ω has the prominent impact on the movement of the particles in the search space. It has first been presented by Shi, Eberhart [44]. In the beginning ω was assigned static value 1.0. Later on, Shi and Eberhart [45; 46] presented a dynamic strategy for adjusting the value of ω. Best results have been recorded for the ω in the interval $[0.8, 1.2]$. PSO with linearly decreasing inertia weight (PSO-LDIW) is also proposed by Shi Eberhart [46]. Basically, inertia weight is clamped to the previous velocity to control its influence for further iterations.

   (d) **Constriction Factor:** The idea of constriction factor is first proposed by Mau-

ric Clerc [47]. It has been shown that the contribution of constriction factor improves the overall performance as well as eliminates the V_{max} from main structure of the algorithm. Several models have been proposed for constriction factor. The following model has been widely used by the researchers.

$$v_i(t+1) = \chi(\omega v_i(t) + c_1 r_1(pbest_i(t) - x_i(t)) + c_2 r_2(gbest_i(t) - x_i(t))) \tag{2.4}$$

$$\chi = \frac{2}{|2-\varphi-\sqrt{\varphi^2+4(\varphi)}|} \tag{2.5}$$

$$\varphi = c_1 + c_2 = 4.1 \tag{2.6}$$

Eberhart has further investigated the performance of the constriction factor in comparison with inertia weight [48]. The improvement in the convergence rate has been observed. However, caused failure in some cases, but the overall performance was encouraging.

2. **Hybrid PSO:** PSO algorithm is capable of solving the problem individually on its own, but sometimes, we use PSO in combination with other techniques. The aim is to attain maximum performance in respect to the desired objective. Robinson and Juang [49; 50] have used PSO in combination with GA as a hybrid technique. Valdez has used PSO with fuzzy logic and GA for neural networks [51]. PSO has also been used in combination with ant colony optimization (ACO) [52]. Zhang [53] has developed an amalgam of differential evolution (DE) and PSO. Some other additional parameters have been introduced to form PSO as a hybrid algorithm [54; 55].

3. **Topological Improvement:** This part of PSO area is related to the work done by other researchers for topological constructions. The improved topological construction of the population aim for the swarm diversity, which further avoids premature convergence [56; 57]. Kennedy [58] has proposed various topological structures. The common topological models are Gbest Model, Lbest Model or ring topology, Wheel, and Von Neuman Model. Each model has its own merits and demerits. Mendes and Kennedy developed a fully informed PSO version [59]. FIPS is focused on the neighbourhood best strategy rather than Gbest and Pbest. It means that the structure is simpler, which is also the aim of many researchers to maintain the simplicity of PSO algorithm. Liang has developed comprehensive particle swarm optimization (CLPSO) [4], which has also the similar approach of the neighbourhood best for updating its trajectories.

4. **Multi Swarm PSO:** The fourth category describes the multi-swarm PSO algorithms. In the multi-swarm PSO, the swarm shares historical information among other participant swarms. As described in dynamic multi-swarm PSO (DMS-PSO) [60]. Another multi-swarm PSO named cooperative PSO (CPSO) is proposed by Berg [61]. Large scale problems are further divided into sub-problems. Thus, CPSO and some other variants have contributed the best solution to the large scale problems and also a new variant to the area of multi-swarm [62; 63] has been developed.

5. **Simplified and Robust PSO:** This category is about the efforts made by many researchers to make PSO algorithm work effectively in the limited resources of computation and memory. In contrast to the above mentioned categories, this category is designated to explore the variants of PSO, which proposed simplification and inexpensive computation requirements. No additional parameters or techniques have been adopted. James Kennedy has introduced a variant of PSO [64] named as Barebones PSO (BbPSO) dependent on certain probability. The concept of Gaussian distribution is used for position update as an alternative to the basic velocity update equation. Zhou has introduced a simplified variant named intelligent particle optimization (ISPO) in [65]. Instead of initializing a swarm, just a single particle is used for optimization process. Iacca has described more explanation and applications of single particle optimization in [66]. A new variant of PSO named fitness evaluation strategy for particle swarm optimization (FESPSO) is introduced [67], where the objective is to find the optimum with minimum number of function evaluations.

After the classification of the literature, we have found that it is very important to explore and contribute further to each category. Nevertheless, the work in this thesis is more focused on the first category of parameter adaptation and also contributes to the other categories to some extent. As the main problem of PSO algorithm is its sensitivity to the adjustment of its parameters, we have proposed generalized and efficient strategy for parameters control as an initial objective of our research. By using Markov chain, it has also contributed to the second category of hybrid PSO [8]. Additionally, the switching technique contributes to the topological enhancement category. Due to its systematic structure, it contributes to almost all categories. The extensive empirical work validates the robustness of the method in competition to the other methods used. This research is based on some variants of PSO algorithm. We will make some brief discussions about those variants in the following section.

2.5 PSO Variants

PSO has been adopted widely in solving real world practical problems because of its modest theory and efficient performance. Therefore, the PSO has become more interesting and important optimizer for researchers. Kennedy, Clerc [68] and many other researchers have reported their work on the improvement of PSO convergence, stability [69; 70; 71], controlling parameters, topological framework and the introduction of auxiliary parameters [5; 72; 73]. Here we explore the major improvements and some well-known variants as the background for our research.

1. **PSO variants with inertia:** Shi and Eberhart [44] have developed a variant of PSO with time decreasing inertia weight.

$$\omega = \omega_{max} - \frac{(\omega_{max} - \omega_{min}) \times t}{T} \tag{2.7}$$

where t represents the index of the current iteration, T is the maximum number of iterations, ω_{max} represents the maximum inertia weight and ω_{min} is the minimum inertia weight having the values between 0.9 and 0.4 respectively. In [74] Shi and Eberhart have developed a new variant named adaptive PSO with fuzzy inertia weight. The weight of inertia is changed according to the best value in the current evaluation and the inertia weight used for that particular evaluation is recorded. So, in the next evaluation that particle inertia weight will be adopted. Later on, PSO with constriction factor is developed by Clerc in [47; 68] which has already been explained in the previous section. This work is based on global PSO version with inertia weight parameter [44].

2. **PSO variants with acceleration coefficients:** In PSO algorithm the particle's movement is modelled by velocity update equation. In the velocity update equation we have further three parts. The first part is related to inertia weight or momentum of the particle, the second part is personal or cognitive influence of the particle and the third part is concerned with the social or global influence of the particle. As we know that each part has an essential influence and control in the success of the algorithm. In the basic PSO [36] a value 2.0 is used as constant for acceleration coefficients and many researchers have adopted this constant value. The concept of ad-hoc weights of acceleration coefficients is introduced by Suganthan [56]. A specific and temporary values are assigned to the acceleration coefficients c_1 and c_2. A new variant PSO with time varying acceleration coefficients named Hybrid-PSO-TVAC

is presented in [43]. A maximum and minimum boundary is given for c_1 and c_2 then program the starts with small values of c_2 for global exploration and big values for c_1 to avoid local convergence.

$$c_1 = (c_{1l} - c_{1f}) \times \frac{maxiter - iter}{maxiter} + c_{1f}, \tag{2.8}$$

$$c_2 = (c_{2l} - c_{2f}) \times \frac{maxiter - iter}{maxiter} + c_{2f} \tag{2.9}$$

where $c_{1f} = 2.5$ and $c_{2f} = 0.5$ are the starting values, and $c_{1l} = 0.5$ and $c_{2l} = 2.5$ are the ending minimum values adopted by cognitive and social learning accelerating coefficients respectively. The weights of acceleration coefficients are linearly increased or decreased with time t, till T, where t is the current iteration and T is the number of maximum iterations.

3. **Hybridized Variants of PSO:** Some evolutionary techniques have been used in combination with PSO algorithms in the hybrid manner. Genetic Algorithm (GA) is used in [75] and selection mechanism is introduced. Juang in [50] has also used GA in combination with PSO algorithm for ANN training. Furthermore, crossover and mutation has also been used in the PSO environment [76; 77] respectively. Zhan has explored many hybrid PSO variants in the background literature and he has also proposed a new variant APSO in [7], where a new adaptive strategy for acceleration coefficients selection has been used. In addition, four states strategy has been developed on the bases of evolutionary state estimation (ESE). A specific value of acceleration coefficient is assigned according to the particle's current state. First of all, the current state is determined by using the evolutionary factor. The mean distance of each particle from the global optimum and each particle is calculated. The fuzzy membership approach is used to measure the degree of membership of each particle to be in a specific state. So, the higher the value of evolutionary factor, the far the particle is from the global optimum and thus the larger value for acceleration coefficient is assigned to jump high to reach the optimum. In other words, the large values of evolutionary factor represents the particle is far away from the global optimum and needs more steps to get to the global optimum. Some limitations have been found in APSO due to fuzzy classification method. To further improve APSO algorithm, a new switching particle swarm optimization (SPSO) is proposed [8] and the Markov chains is used for velocity update. The SPSO algorithm is based on the similar concept of APSO [7]. Basically, the SPSO has been developed to overcome the shortcomings of the APSO algorithm by introducing Markov chain instead of fuzzy classification. The similar concept of evolutionary factors and four state is used here.

SPSO has been used in many applications and has shown satisfactory performance in comparison to other algorithms. The performance of SPSO algorithm has examined on some mathematical benchmark functions and then it has been used for parameters identification in genetic regulatory networks (GRN). Due to its promising performance many researchers have applied SPSO successfully in many other optimization problems. Recently, [78] has applied SPSO for face recognition. SPSO has been adopted for parameter estimation in [9] of lateral flow immunoassay. The same algorithm has been modified with local evolutionary by Zeng in [79] and has further been used for quantitative analysis in combination with differential evolution.

4. **Proposed Variants:** The background study of PSO algorithm has motivated to move in this direction and make some further improvement to this area. Therefore, an N State Markov Jumping Particle Swarm Optimization (NS-MJPSO) is proposed in this work. A more robust and sophisticated structure for state switching is proposed. An N states concept is used instead of four state. An appropriate value of N state is assigned according to the nature and size of the problem. It also depend on the objective of the problem. For instance, if accuracy and better solution quality is required then we assigned large value to N states. As we know that the larger the N the accurate the results are. However, computation burden is increased with larger N states. In the N states, the four states are further divided into sub-states. It means that we can improve the quality of accuracy by describing each states in small parts, where each sub-state represents the degree of association to a specific state. In other words, to what extent the particle has reached a particular region. The rest of the structure is exactly the same as basic SPSO. In Chapter 3, the development of NS-MJPSO has briefly been explained with formulation and experimental work using mathematical benchmark functions. As we know that the increase in N states causes extra computational burden in the novel NS-MJPSO algorithm and also it is sometimes difficult to assign jumping probability accurately. For that reason, we have proposed another novel PSO algorithm named N states switching particle swarm optimization (NS-SPSO) algorithm, which excludes the Markov chains method. The novel NS-SPSO is purely based on the evolutionary factor information, which has significantly minimized computational burden and algorithm complexity. In the proposed NS-SPSO, we don't need domain knowledge. The N states are described as the sub-states or stage of the four main states. The velocity update is entirely dependant on the evolutionary factor, where N number of acceleration coefficients are pre-initialized according to the sub-states. The novel NS-SPSO has been tested for various mathematical benchmark functions. It has been shown in the tabular and

graphical illustrations of simulation results that the proposed NS-SPSO has the advantage of adaptation due to the evolutionary technique. The proposed NS-SPSO is the first best algorithm in terms of the shortest computation burden and second best in terms of accuracy of the solution.

In this next section we discuss about some applications of PSO reported in the literature.

2.6 PSO Applications

1. **Artificial Neural Network Training (ANN):** We have already described briefly the applications of PSO algorithm in many disciplines. One of them is the training of neural network that has investigated by many researchers. The improvement to the strategy of synaptic weights and topology have been selected as the main interest in ANN. Both of the supervised and unsupervised learning approaches have been considered. The authors in [80; 81] have reported the comparison of PSO algorithm with back-propagation algorithms in ANN, where the faster convergence speed has been observed by using PSO algorithm. PSO was first adopted for training the Recurrent Neural Network Model in [82]. PSO algorithm has also been used for nonlinear optimization problems, SVM, radial basis functions and fault detection in [83; 84; 85].

2. **Digital Signal Processing:** PSO has been used for digital-filtering and the results are then compared with GA, where PSO was found more efficient than other methods [86]. In [87] PSO algorithm has been used for signal detection, multi-user detection and estimation [88; 89].

3. **Image Processing:** In image processing, PSO algorithm has been used for image segmentation [90; 91], image classification [92; 93], noise cancellation [94], clustering [95; 96], image restoration [97; 98], pattern matching [99] and texture analysis [100]. Given by all the above literature, PSO is one of the efficient, simpler and robust method for image processing.

4. **Robotics:** The researchers in the field of robotics have also applied PSO algorithm for fuzzy network design in robotics [101; 102]. PSO has been used in robotics path planning and source localization [103; 104].

5. **Power System:** Yamille [105] and AlRashidi [106] have summarized an extensive survey of PSO variants and their applications in power systems. The applications of

PSO algorithms in the area of power systems depends on the nature and the objective function of the optimization problem in hand. Those problems are classified as linear and non-linear, constrained and unconstrained, integer and mix-integers. While some problems require differentiation, derivative based methods are applied in that case. Due to the adaptive nature of PSO algorithm, it has been slightly modified and then applied to the variety of optimization problems in power systems. The most common problems where PSO has been applied are the optimal problem flow [41; 107; 108], reactive power dispatch optimization[109; 110; 111], Power load forecasting [112; 113], Controller design of power systems [40; 114], transmission network planning [115; 116], power generation planning [117; 118], losses minimization [119; 120] and the power system operations problems [10; 121; 122]. PSO algorithm has similar implementation procedure for all types applications, but the only thing that needs to be formulated is the objective function and the only difference in all the applications is the parameter adjustments. Power system operation is the focus for applications in this thesis. In the following subsection we will explore the background for economic load dispatch (ELD) and dynamic economic load dispatch (DELD) using basic PSO algorithm and other variants of PSO algorithms.

2.6.1 PSO Applications in ELD Problems

We have further categorized the literature into two subsections. In the first subsection, we have summarized the applications of traditional PSO algorithm and in the second section we have described the modified PSO applications in ELD problems.

2.6.1.1 Applications of Traditional PSO in ELD

ELD has further been divided into two sub-categories. The first one called static economic load dispatch, which is the simple case of ED problems and the second one is the dynamic economic load dispatch, which is very complex and highly nonlinear case of ED problems. We will briefly discuss both of them in Chapter 5 and Chapter 6 respectively. Traditional PSO algorithm has been used for many optimization problems in power system. Whereas in the ELD problems, the traditional PSO has first been proposed by Gaing in [10]. The generators have many nonlinear constraints, which are denoted as power balance, generation limits, prohibited operating zones, line losses and ramp-rate limits. The traditional PSO algorithm has been used to evaluate the quadratic cost function or objective function. The $6-units$, $15-units$, and $40-units$ benchmark data is used. The results are

then compared with the GA for the same objectives. The traditional PSO has produced good quality solution with minimum cost value and shortest in computation time. In [15] Gaing has applied traditional PSO algorithm for dynamic economic load dispatch (DELD) problem. As we know that, DELD is highly complex and large scale problem, where the power load changes dynamically over a period of time. Moreover, the constraints are highly nonsmooth or nonlinear applied to each individual unit. It means that the DELD problem needs to be solved with the minimum generating cost and within the satisfaction of the constraints boundaries. In the given literature DELD is solved for 24 hours, where different load has been selected for each hour. The cost minimization function is similar to the static ELD problem, but the only difference is involvement of dispatch hours t in case of DELD. All the constraints mentioned in ELD are considered here along with the new constraint spinning reserve used in [123].

In [124] the authors have used the PSO with constriction factor for DELD problem. In [121] the authors have proposed a novel approach for solving ELD problem with valve-point loading effects and multi-fuels constraints. These constraints make the cost objective function non-smooth. In [125] the authors have applied the basic PSO algorithm to various ELD problems considering all generator constraints including multi-fuels, multi-area and environmental constraints. Environmental constraints have imposed the law about pollution reduction. It is obvious that the increase in the power demand leads to the increase power generation, which on the other hand causes increase in pollution. For that reason, research in the domain of green energy or renewable energy is the important topic. The satisfactory performance of the traditional PSO in ELD and DELD problems has encouraged researcher to further modify and enhance the structure of PSO algorithm, which can further be used for ELD and DELD problems. Further, in the next section we have explored the applications of modified versions of PSO algorithm.

2.6.1.2 Applications of PSO Variants in ELD

Some additional techniques are used in combination with PSO algorithm to get the maximum performance from the objective. In [122] the modified version of PSO algorithm with Gaussian probability distribution has been developed. The Gaussian approach has first been proposed in [126], where the random numbers in the velocity update equation named as $rand_1$ and $rand_2$ are replaced with the Gaussian probability distribution (GPD) parameters. The random numbers in the interval $[-1, 1]$ are created by applying Gaussian probability distribution. The values are mapped with the basic interval of random numbers $[0, 1]$ after creating the GPD. Moreover, a technique called chaotic sequence in also used here to avoid premature convergence to the local optimum. The proposed algorithm have

been tested for $15-units$ and $20-units$ in comparison with traditional PSO algorithm and some other variants, where the proposed algorithms performed better than others.

Another variant of PSO algorithm called New PSO (NPSO) has been proposed in [127]. The modification in the NPSO version is two fold. The first one is the split-up mechanism in c_1 the cognitive parameter of PSO algorithm. Normally, we record the best position ever visited by each particle, while the worst individual position is recorded as well and c_1 is further divided into c_{1a}, and c_{1b}. Best exploration is achieved by adding this mechanism. Furthermore, an additional technique namely local random search (LRS) is combined with the novel NPSO for better exploitation in the search space. The initial population is generated according to the modified LRS approach. This NPSO is applied to the non-convex ELD problems along with LRS mechanism. The NPSO-LRS is then tested for three different unit systems having $6-unit$, $40-unit$ and $10-unit$ systems with various operational and system constraints corresponding to each unit of the system. The results obtained from the proposed have been compared with other existing methods. It has been observed that the proposed algorithm performed better than other methods. Later on, the authors of this paper have proposed another version of PSO algorithm named anti-predatory PSO algorithm with applications in ELD problems [128]. In [11] an improved PSO algorithm (IPSO) is proposed. The method of chaotic sequences and crossover is is employed. The modified version is then applied to very large-scale ELD problems in power system. The operational constraints valve-point loading, multi-fuels, equality and inequality constraints, with and without power losses, prohibited operating zones are considered for ELD problem. 10, 15, 40 and 140 unit system data is used. It has been concluded at the end that the proposed algorithm has performed very well in comparison to the others methods. A new hybrid version of PSO is introduced in [129] with applications in ELD non-smooth problems, where the successful performance have been reported. In the next subsection we describe the literature about dynamic economic dispatch.

2.6.1.3 Applications of PSO and Variants in DELD

As we know that, the DELD problems have the dynamic nature due to the large-scale fluctuating load demands with time. Subsequently, with the formulation of dynamic dispatch problem the research has been motivated to several optimization methods incorporating the characteristics and the capability to solve the problems with constraints. The first attempt to find a competent method for solving such kind of problems was based on the mathematical methods such as gradient based method, lambda iteration, Lagrange multiplier based method, interior point method, dynamic programming and non-linear programming. As mentioned in the previous section DELD is a complex, high dimensional and non-linear

optimization problem. DELD ensure the supply of power to the dynamically changing loads during the period of 24 hours. The dynamic and non-linear constraints such as, ramp-rate limits and valve-point loading effects makes the objective function non-smooth and difficult to solve. The analytical and traditional methods fail to work due to curse of dimensionality and non-linearity in objection functions. Hybridized and enhanced versions of current optimization methods have been used to solve DELD problems.

In [130] the authors have applied an improved version of PSO algorithm. Chaotic mutation has been used to avoid premature convergence to the local optimum. This method enhances the global search ability. The acceleration coefficients c_1 and c_2 have been computed and controlled before objective function evaluation. The value of c_1 is initialized with a small value and thus increased in each iteration with ratio of maximum iterations and the number of particles used. The social acceleration factor c_2 value is decreased iteratively with the same ratio of c_1. The proposed IPSO algorithm is then applied to DED objectives of having 10 and 30 unit systems. The results show the better performance of IPSO in solution quality, computation time and convergence as compared to the other algorithms. In [131] the authors have introduced a novel improved version of PSO algorithm. In this proposed algorithm the feasibility and probabilistic based with priority list based method to deal with non-linear type of constraints. The best thing in this method is that it does not need any penalty parameters for handling the constraints. The proposed algorithm is then tested on three case studies of 10 unit systems with and without considering transmission losses into account in the first two cases, while in the third case study the 10 unit system data is used in the triplicate manner and hence a large-scale 30 unit is obtained.

In [18] the authors have proposed a new version of PSO algorithm namely adaptive PSO with successful applications in ELD and DELD. In the proposed APSO, the movement of particle is adjusted according to the current position of particle, if the particles are in the best region the acceleration is very slow. The ranking mechanism is adopted and the first rank is assigned to the best particles. The proposed algorithm is tested on the various unit systems for ELD and 5 unit system of DED problems and has produced best results in comparison to the other algorithms. Gaing has applied traditional PSO in [15] for DELD problems. Previously, Gaing has used the same algorithm for static ELD problems. In the scheduled time t for dispatch to the load according to the minimum and maximum generation limit of each generator. The minimum per unit cost F_{min} and maximum per unit cost F_{max} have been computed. The population is initialized in the range of F_{min} and F_{max} along with the dimensions of the unit system. The initialized population satisfies all constraints of power balance and prohibited operating zones. The transmission losses are calculated on B coefficient matrix. Then the evaluation function is calculated for all

particles, during the maximum number of iterations.

The proposed method has been tested on two case studies of 6 and 15 unit systems. In the first case study of the 6 unit system for 24 hours dispatch with the dynamic changing loads in the interval of 930 Mega watts and 1263 Mega watts. In the second case study of having 15 unit system for the period of 24 hours and dynamically changing loads in the interval 2215 Mega watts and 2953 Mega watts. The simulation results have demonstrated the capability of the proposed algorithm for the most complex optimization problems.

Artificial intelligence based techniques have been widely used individually and also in the hybrid manner for solving DELD problems. In [13] the authors have developed a hybrid multi-objective PSO algorithms namely fuzzy self-adaptive learning PSO algorithm (FSLPSO). The inertia weight parameter is controlled by fuzzy adaptive mechanism. The acceleration coefficients are assigned fixed values 2. The fuzzy adaptive learning concept is used here because of the high-level uncertainties. The first uncertainty is the load demand that can be changed with time. The second uncertainty is the renewable power system resources of wind. The problem is formulated on scenario-based method to deal with such kind of uncertainties. Therefore, an appropriate scenario has been adopted for a specific scheduled time and a desired load. The emission is considered as an objective to be minimized along with the total energy saving. Subsequently, it produces a multi-objective task for the proposed algorithm and the proposed algorithm has performed very well for dynamic economic and emission dispatch (DEED) problem. The constraints applied here are valve-point loading effect, spinning reserves, ramp-rate limits and also the constraints concerned with wind power plant. The 10 unit systems with and without losses, where the 100 unit system without losses are used for the experiments.

According to [132] the DED problem has been investigated considering its operational constraints for 5, 10 and 110 unit systems using Lbest PSO. The previous literature support the feasibility of PSO algorithm of ELD and DELD problems, though, their limitations have also been reported. Producing the suboptimal solution is one of the main problem. The algorithm get trapped into the local minimum. To avoid this problem a dynamic varying sub-swarm strategy is introduced in the Lbest version of PSO. The entire population is further divided into sub-swarms and the sub-swarm are evaluated individually. The best position found in each sub-swarm is compared and thus the G_{best} has been selected after comparison. The proposed modified algorithm has produced the best value for the objective in comparison to the other algorithms solving the same problem.

In [133] DELD is formulated as non-linear and non-convex optimization problem because of the valve-point loading effects in the system. A novel enhanced adaptive PSO algorithm is developed and then applied to the DELD problem. The dynamic characteris-

tics of the system are the ramp-rate limits and calculating the losses simultaneously. The novel EAPSO has the ability to deal with the problems of the existing traditional PSO algorithms or its variants particularly it avoids premature convergence. This is because of the mutation concept adaptation, dynamic inertia weight and also the self-adaptively adjusted acceleration coefficients. The novel EAPSO has been tested on 5, 10 and 30 unit systems of the power plant. The results has shown that the proposed EAPSO outperformed the algorithms in comparison.

Another, improved version of PSO algorithm namely TVAC-IPSO has been developed in [134]. The primary objective of DED is to ensure the supply of power to the loads for a given time span in the minimum generating cost considering the system and operational constraints. The iteration based PSO (IPSO) [135] has been first proposed and used for ELD problems. In this paper IPSO has further been modified by adding the time varying acceleration coefficients concept. The iteration best or time best particle is recorded and new terms L_{best}, c_3 and r_3 are introduced in velocity update equation. This new algorithm is then applied to two case studies, 5 and 10 units of dynamic economic dispatch problems. The non-linear, dynamic and non-convex kind of constraints such as valve-point loading effects, ramp-rate limits, prohibited operating zones and transmission losses were considered before solving the objective. In the next section we discuss the applications of the new switching PSO algorithm. Furthermore, we have modified the switching PSO and two novel enhanced versions of SPSO have been developed.

2.7 Applications of Switching PSO

In [8] the authors have first proposed a novel switching PSO algorithm. The Markov chain concept is applied as the base for this algorithm. The main concept of this algorithm is derived from the adaptive PSO (APSO) [7]. Both of the algorithms have used evolutionary factors, population probability distribution and four states strategies. For the evolutionary factor E_f the mean distance is calculated by the Euclidean formula.

$$P_d(i) = \frac{1}{N-1} \sum_{j=1, j\neq i}^{N} \sqrt{\sum_{k=1}^{D} (x_i(k) - \bar{x}_j(k))^2} \tag{2.10}$$

where N and D represents swarm size and dimensions of the problem respectively, and then E_f is calculated as follows.

$$E_f = \frac{P_{dg} - P_{d(min)}}{P_{d(max)} - P_{d(min)}} \in [0, 1] \tag{2.11}$$

where P_{dg} represents the index of global best particle $P_{d(max)}$ and $P_{d(min)}$ represent the maximum and minimum mean distance respectively. Hence, after finding the values of the E_f the fuzzy classification is derived in APSO and Markov jumping is used in SPSO. The four states are *Exploration*, *Exploitation*, *Convergence* and *Jumping-out* states. Some limitations of APSO have been highlighted and thus some modifications have been made to develop an enhanced version. This version is named as Switching PSO (SPSO). Here the first state is considered as *Convergence*, second *Exploration*, third *Exploitation* and fourth as *Jumping-out* state. A new mode-dependent switching parameter is introduced here. A suitable value for acceleration coefficients is assigned according to the current state of the particle. A smaller value is assigned in convergence state, while a larger value in jumping state. The weight of inertia ω is also dependent on evolutionary factor and current state of the particle. An adaptive value of ω is assigned in each iteration according to the current state of the particle. The SPSO has been tested on several mathematical benchmark functions and has also been applied to the genetic regulatory networks (GRN) for parameter identification. Recently in [78] SPSO has been added the concept of Wavelet Neural Network (WNN) and then applied parameter optimization in face direction recognition. According to [136] SPSO has been modified and applied in the hybrid manner by introducing differential evolution (DE) and the new algorithm is then applied to lateral flow immunoassay analysis. In [9] the SPSO has been used with hybrid EKF for parameters estimation in lateral flow immunoassay. Consequent to the extensive search for the applications SPSO, we have found very limited number of applications. The idea of Markovian jumping and switching mechanism has widely been used in other techniques. After having the extensive search and study of the literature, we have noticed that SPSO has never been applied in power systems.

In this thesis, we have further investigated the performance of SPSO in comparison to the other well-known PSO variants. Through these investigations, we have made some assumptions for the improvement of performance after having some further modifications in the basic version of SPSO algorithm. Some assumptions have been made for the improvement in the performance of basic SPSO algorithm. Two new versions of SPSO have been proposed. In the following section we discuss the newly proposed algorithms and their application in ELD and DED problems in power systems.

2.8 NS-MJPSO, NS-SPSO and Their Applications in ELD and DED Problems

The newly developed SPSO has exhaustively been analysed for further improvement. Subsequently, the idea of N state is proposed with the assumption of the best performance and quick convergence. The proposed algorithm is named N State Markov Jumping PSO algorithm. Previously, in the basic SPSO the four state strategies were used as explained in the previous section. A generalized and automatic controlled strategy for state-switching is proposed here. Basically, the four state term sounds like more specific to a certain capacity problem. However, the N State is more generalized term and it can take any value. Here, we have used 4, 6 and 8 in our experiments. The four states are further divided into sub-states. In the relatively high dimension search space the sub-state represents the degree of being in a particular state. By this mechanism we assume that the algorithm will converge to the global optimum for following the smooth curve. As we know that by increasing the number of states, the accuracy of the solution increase and the computational burden slightly increases. For that reason, we will first test for the small values and then gradually on the large value of N. The N value having the small evaluation error and shortest computation time will be considered. As we mentioned previously that the basic SPSO has never been applied in power system. So, the first contribution of this thesis is the implementation of new algorithm in very important optimization problems of power system. The feasibility of SPSO and the proposed N State Markov Jumping PSO is elaborated as follows:

The concept of Markov chain has widely been used for economic and financial, decision making, and control systems [137; 138; 139; 140]. The Markov chain has also used in the power system related applications [141; 142] for unit commitment and economic dispatch problems. Transition probability distribution is drawn for all states. The Markov chain might preserve its current state or it could change to another state corresponding to its transition probability. Basically, transition is the process of switching within the various states. This switching is dependent on its transition probability. It is already known that the basic PSO has the problem of getting trapped into local optimum. The Markov chain is used in the velocity update equation for the switching or jumping purposes, which has enabled the swarm to perform global search. The parameters are assigned the value according to the current state. Subsequently, the swarm converge to the global optimum in few steps. The Markov Jumping is applied to population distribution determined by evolutionary factors. Evolutionary factor is further explained in the next paragraph.

The evolutionary factor E_f has been used to optimally control the process of the PSO algorithm. In the algorithm the swarm diversity is not only influenced by the increment of iteration, but the evolutionary factor has also controlled over the population distribution. As the population is initialized the whole population get scattered in search space. Though, the particles move towards their best neighbours and group together in the local or global region of the search space. These groups or clusters were then analysed by to determine the state of each particle. After the encouraging performance of evolutionary techniques applications in numerous backgrounds the researchers have combined evolutionary technique with PSO algorithm [6; 143]. The evolutionary algorithms has been applied to numerous backgrounds including power systems, and it has produced promising results. In [144; 145; 146] evolutionary techniques have been used for the economic dispatch problems in power system.

There is always a room for improvement in the research particularly in the power systems. Since the substantial increase in the daily energy usage the research in power system is focused on the balanced supply of power to the loads at the minimum cost. The research in this area is an inadequate to the increasing demand on the other hand. Power system is a huge system, which needs huge planning, massive budget, extensive technical expertise and long time to start running. Meanwhile, the existing systems are focused to be maintained and operated in the optimal manner. Several strategies for efficient scheduling and economic operation of the existing power systems were proposed in the literature. In this thesis, we aim to contribute an improvement in this area. We have studied and analysed the hybrid type algorithms, the evolutionary and Markov chain techniques in combination with PSO algorithms for ELD and DED problems in power systems. In Chapter 3, we present our first proposed algorithm. The proposed algorithm efficiently explores the search space and converges to the global optimum very fast. However, the computation burden increases with increasing the number of states. To solve the problem of computation time, we have proposed another algorithm in Chapter 4, the second proposed algorithm. Furthermore, in Chapter 5, we have applied both of the novel algorithms to static ELD problems and in Chapter 6, we have applied both of the algorithms to DELD problems.

Chapter 3

A Novel N State Markov Jumping PSO

3.1 Introduction

In this chapter we introduce a new variant of Particle Swarm Optimization algorithm with N State Markov Jumping (NS-MJPSO). The aim of this method is to improve the global search performance, and its capability of solving the real-world problems. Basically, the real world problems are non-linear and that's a challenging problem to be solved. It has been revealed in the literature that evolutionary types of methods have a better performance in non-linear problems. The wide use, simple structure, and easy implementation of the Particle Swarm Optimization (PSO), are the motivation for this work. PSO is a competent, population-based, swarm intelligence technique for optimization. The traditional PSO along-with its modified variants have been explored and re-evaluated. The problem of getting stuck in the local-optimum is observed in PSO evaluation. However, in the NS-MJPSO, we combine PSO algorithm with Markov chain. Markov chain is used to keep the particle moving in the search space with certain probability. The second reason for using Markov chain is also its better performance applications in economic-based, decision and control systems. The proposed method is examined by some widely used, uni-modal and multi-modal mathematical benchmark functions. The evaluation results are then compared with the most cited existing state-of-the-art PSO variants on the same functions. Later this chapter, we present some background work in Section 3.2, the structure of traditional PSO in Section 3.2.1. In Section 3.3, the proposed work is briefly described. The experimental work is given in Section 3.4 and In Section 3.5, we have summarised the whole chapter.

3.2 Some Related Work

Particle Swarm Optimization (PSO) is a population based method developed in 1995 by Kennedy and Eberhart [36; 37]. The idea is taken from the swarm's agents intelligent learning behaviour such as birds flocking and fish schooling [147]. PSO uses a simple mechanism of copying the best location find their peers, the particles compare their current states value to the neighbours and then jump to the new location if it is better than their current state. In other words, the particle memorises the location find by its own experience known as the personal best position P_{best}. It is also called the local best position or the best position find by each particle so far. The objective is to find the best position in entire search space. The best position in the entire swarm and for all P_{best} is called the global best position G_{best}. It is the best position find by the entire swarm. PSO algorithm is described as velocity and position update. Two acceleration coefficients named as cognitive c_1 and social c_2. Initially, a constant value 2 is used for both of the acceleration coefficients.

According to the basic PSO algorithm, two uniformly distributed random numbers in the range of $[0, 1]$, denoted as $rand_1$ and $rand_2$. The simple framework, easy implementation, few parameters adjustment and fast convergence are the impressive attractions in the PSO algorithm. That's why PSO has been used and modified by many researchers in the last two decades for various real world optimization problems [1; 2; 5; 73; 148; 149; 150].

PSO algorithm is briefly studied and analysed for further improvement and solving the current problems that has been observed during its applications in the real-world problems. The random values causes instabilities and inconsistencies in the evaluation results. These inconsistencies have weaken the performance of PSO algorithm, particularly in the multi-modal and high-dimensionality problems [4]. Similar to the other evolutionary algorithms, PSO also sometimes get trapped into the local optimum, called premature convergence. The excessive function evaluations (FEs) are required to escape from the local optimum [3]. To address all the above mentioned limitations, many variants of PSO has been developed. Shi and Eberhart [45], have introduced constant inertia weight ω with two acceleration coefficients. This version is again further modified as an adaptive ω parameter in [46]. Fast convergence has been achieved by using the constant and adaptive inertia ω. Another variant has been developed by Shi and Eberhart in [74], where a fuzzy strategy is proposed to control the velocity by dynamically updating the ω. Furthermore, various techniques have been proposed by for the efficient adjustment of acceleration coefficients [43; 151]. c_1, c_2 and ω are assigned the maximum and minimum value for each. The final values are calculated according the maximum number of evaluations denoted as Time-Varying Coefficients. The purpose of TVAC version is to control parameters and converge quickly to global optima. Some supplementary parameter are applied to help the basic parameters efficiently find their values.

Furthermore, to speed up convergence and to escape from local optima have become the two utmost significant research topics in PSO. Numerous works have been done to attain these goals [1; 2; 3; 4]. In this improvement, controlling the parameters of algorithm, the addition of supplementary search operators and enhancing the topological structure have become the three most prominent approaches [5]. It has found to be challenging to achieve both the goals simultaneously. The comprehensive-learning PSO (CLPSO) [4] emphases on solving the problem of sub-optimal solution. However, the algorithm results in slower convergence.

Moreover, in the developmental studies of PSO algorithm the primary objective that has been considered is the minimization of computation complexity. Secondly, the problem of local optima or premature convergence has been investigated exhaustively by [1; 2; 3; 4]. The improved algorithms have been further tested by doing some applications in real-

world environment. Due to the uncertain, non-linear and complex nature of the real-world problems there is always a gap for improvement in the current dealing algorithms. In response to that, the secondary techniques have been applied to control the parameters significantly [6; 7; 8]. The topological structures have been developed to achieve the second objective to ensure global optimum and avoid premature convergence [4].

Adaptive PSO (APSO) [7] has been focused on achieving both the goals by introducing the evolutionary factor E_f, which has further added the population distribution characteristics, the mean distance between the global best and other particles in the swarm. Four states S_1, S_2, S_3, and S_4 have been described by taking population distribution information E_f in to account, which symbolizes convergence, exploration, exploitation and jumping out states respectively. Fuzzy classification method has been used, which has resulted some limitations of excessive computation acceleration coefficients in each generation, swarm stagnation in the local optima, if the current global best is the local optimum and the last one is the complicated implementation of classification method.

Switching PSO (SPSO) in [8] has further improved the performance of the APSO and has addressed its limitations. A novel approach, where the Markovian switching has been used to avoid getting trapped in to local optima (premature convergence) and leader competitive penalized multi-learning approach (LCPMA) is used to improve global search performance [9]. Same four states as described in [7] have been used here. However, a new mode-dependent technique with Markovian Jumping parameters has been adopted in the velocity update equation to improve the search performance. SPSO has produced promising results in [8; 9]. SPSO has been re-evaluated in comparison with some other state-of-the-art algorithms on some benchmark problems. It has been observed that SPSO a very problem specific and requires much parameter adjustment to perform better in the dynamic environment or any other new problem. Therefore, to further enhance the performance of SPSO, it has been modified in this thesis.

The proposed NS-MJPSO contribution is two fold. The first improvement is the robust N states concept rather than four used in SPSO, where N takes value $N = [1....N]$. The second is adaptation of linearly decreasing inertia weight (LDIW) [45]. Third is the computational complexity, that has been improved by simplifying the structure. Proposed NS-MJPSO has been evaluated on 12 widely used benchmark functions given in Section 3.4. In the next Section 3.2.1, we have explored the development of PSO algorithms starting from the traditional PSO algorithm.

3.2.1 Traditional PSO Algorithm Structure

In PSO algorithm swarm represents the population of n particles, where each particle i represents a potential solution to a problem in hand and each particle i is composed of two vectors, where the first one is the velocity of ith particle in Dth dimension and t time is represented as $v_i(t) = [v_{i1}(t), v_{i2}(t), \cdots, v_{iD}(t)]$ and the second one is position of the ith particle in Dth dimension, and in t time is represented as, $x_i(t) = [x_{i1}(t), x_{i2}(t), \cdots, x_{iD}(t)]$, where D represents dimension of the solution search space. The swarm velocities and positions are initialized randomly with their respective boundaries $x_{in}(t) \in [x_{\min,n}, x_{\max,n}]$ $(1 \leq n \leq D)$ with $x_{\min,n}$ and $x_{\max,n}$, of the search space. The velocity is limited to a maximum V_{max} is set to 20% of the search boundary [73]. Throughout the evaluation process, each particle i with dth dimension are updated as follows.

$$v_i(t+1) = \omega v_i(t) + c_1 r_1(pbest_i(t) - x_i(t)) + c_2 r_2(gbest_i(t) - x_i(t)) \tag{3.1}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{3.2}$$

where ω is called inertia weight [44], c_1 and c_2 are the cognitive and social learning factors also called acceleration coefficients [37]. Two uniformly distributed random numbers denoted as r_1 and r_2 are generated between $U[0,1]$ [36], where pb represents $pb_i = (pb_{i1}, pb_{i2}, \cdots, pb_{iD})$. Personal best is the position with the best fitness found by the ith particle itself so far and gb represents $gb_D = (gb_1, gb_2, \cdots, gb_D)$, where global best is the best particle ever found by the entire swarm. In the PSO neighbourhood version n_{Best} is used for global best, G_{Best} is used for the global version and L_{Best} is used for the local version of PSO. The particle's personal experience and its social interaction determine iteratively the direction towards its best position. A step-wise Pseudo-code 1 is presented here followed by the basic flow chart in Figure 3.1 using [37].

3.2.2 Developments of PSO

PSO has gained much attention due to its simplicity and effective performance. Numerous work have been done to address all the limitations of PSO algorithm and make it more convenient and reliable. Therefore, various methods have been used in [45; 46], where the PSO with inertia weight ω has been developed to fine-tune the search space. It has been observed that large inertia weight induce global exploration and small values effect local exploitation. A linearly decreased inertia weight (PSO-LDIW) has been proposed in

Algorithm 1 Traditional PSO Algorithm

Initialize the swarm positions and velocities randomly in the search space, with D dimensions
while Stopping condition not true **do**
  for Each particle i **do**
    Update velocity for all swarm using Equation (3.1)
    Update position for all swarm using Equation (3.2)
    Evaluate the fitness function $f(\overrightarrow{X}_i)$
    if $f(\overrightarrow{X}_i) < f(\overrightarrow{P}_i)$ **then**
      $\overrightarrow{P}_i \leftarrow \overrightarrow{X}_i$
    end if
    if $f(\overrightarrow{X}_i) < f(\overrightarrow{P}_g)$ **then**
      $\overrightarrow{P}_g \leftarrow \overrightarrow{X}_i$
    end if
  end for
end while

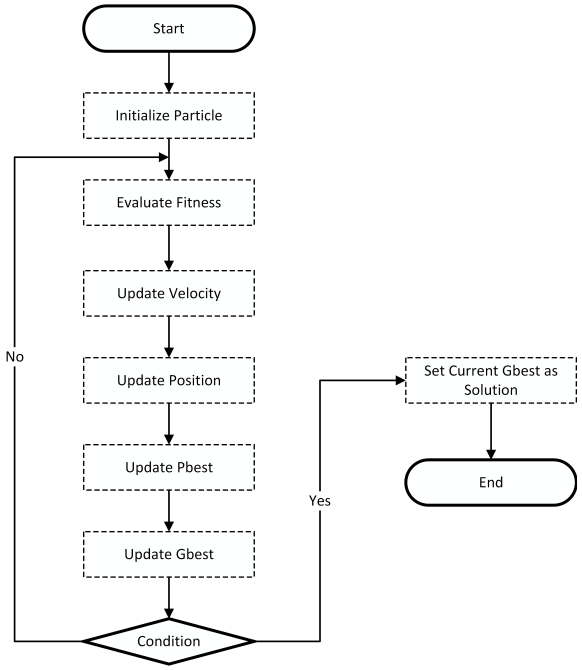


Figure 3.1: Basic PSO Flow Chart

[45; 46] ω is calculated as follows:

$$\omega = (\omega_{max} - \omega_{min}) \times \frac{\text{maxiter} - \text{iter}}{\text{maxiter}} + \omega_{min} \tag{3.3}$$

where $\omega_{max} = 0.9$ has been given the starting, and ω_{min}=0.4 the ending boundary for inertia weight, *maxiter* is the number of maximum iteration, and *iter* is the value of current iteration. Mauric Clerc has introduced the new PSO with constriction factor for analysing and improving convergence behaviour in [68]. The constriction factor equation is given as follows:

$$\chi = \frac{2}{|2-\varphi-\sqrt{\varphi^2+4(\varphi)}|} \tag{3.4}$$

$$\varphi = c_1 + c_2 = 4.1 \tag{3.5}$$

where χ is set to 0.729, c_1 and c_2 are both assigned 2.05 in [68]. It has been shown in [48] that practically, constriction factor is the same as inertia weight. In [44] the authors have focused on the global-version of PSO (GPSO) with inertia weight defined in Equation (3.1).

Furthermore, the other two very important parameters of PSO are acceleration coefficients c_1 denoted as cognitive learning and c_2 denoted as social learning factors. In [152] the authors have described two models, social-only and cognitive-only. It has been shown in the results that the acceleration coefficients have the significant influence in the control of PSO algorithm. In [36] the authors have proposed constant value 2.0 for both of the acceleration coefficients, which has widely been adopted. In [56] it has been presented that the values of acceleration coefficients can be adjusted and tuned according to the problem nature and dimensionality. In [43] another algorithm named PSO with time-varying coefficients (PSO-TVAC). The cognitive c_1 and social c_2 learning factors have been modelled in linearly decreased structure in Equation (3.6) bellow. PSO-TVAC has also the similar concept as the PSO-LDIW [46].

$$c_1 = (c_{1l} - c_{1f}) \times \frac{maxiter-iter}{maxiter} + c_{1f}, \tag{3.6}$$

$$c_2 = (c_{2l} - c_{2f}) \times \frac{maxiter-iter}{maxiter} + c_{2f} \tag{3.7}$$

where $c_{1f} = 2.5$ and $c_{2f} = 0.5$ are the initial values and $c_{1l} = 0.5$ and $c_{2l} = 2.5$ are the ending values adopted by cognitive and social learning coefficients respectively.

Moreover, topological improvement is that has largely been studied by many researchers for improving the search performance of PSO algorithm. In [59] the authors have proposed a fully informed particle swarm optimization algorithm (FIPSO) the entire swarm is con-

sidered as one neighbourhood. Each particle move according to its own knowledge about the entire swarm. Evolutionary operators such as selection [75], crossover [76] and mutation [77] have been added to PSO for topological improvement of the swarm. In [4] a comprehensive learning PSO (CLPSO) has been developed with the similar motivation of topological improvement. It has also described learning behaviour of the particle with the best position found in its history. Later on, an adaptive PSO (APSO) [6] with evolutionary factor and fuzzy membership for the automation and control of its parameters. Similarly, a switching PSO (SPSO) has been proposed recently in [8]. The same four states strategy and evolutionary factor [7] has been followed. However, a new mechanism of Markovian Jumping has been introduced for better search performance as a supplementary parameter. Furthermore, the *leader competitive penalized multi learning approach (LCPMA)* [8] is also adopted for improving the global and local search performance.

3.3 A Novel N State Markovian Jumping PSO

This section presents a novel NS-MJPSO for the improvement of the search performance. Firstly, the value of N state is determined. An auxiliary parameter is added to the velocity update Equation (3.1). Performance in the different number of states is evaluated for 12 uni-modal and multi-modal widely used benchmark functions [4; 153], which are given in Section 3.4, Table **??**.

3.3.1 Population Distribution and Evolutionary Factor

In the beginning of population distribution, the particles are spread out in the search space. In the iterative evolutionary process the particles group together in the later stages and find their local and global optimal places in the search space. Therefore, the extraction of information from the population distribution and using that for further describing the evolutionary state is an important research topic in PSO. Hence, the population distribution information in each generation is important to be recorded. A clustering based technique was introduced for evolutionary state estimation in [6; 143]. Whereas, fuzzy classification method is used for calculating four evolutionary states in [8].

In the first step of the population distribution, where the mean distance from the global best particle in the search space for each particle i is calculated. The particle having small mean distance from the global best is possibly be in convergence state and more likely to stay in the same state. On the other hand, the particle with the maximum mean distance would more likely to change its state and jump to another one. The mean distance is

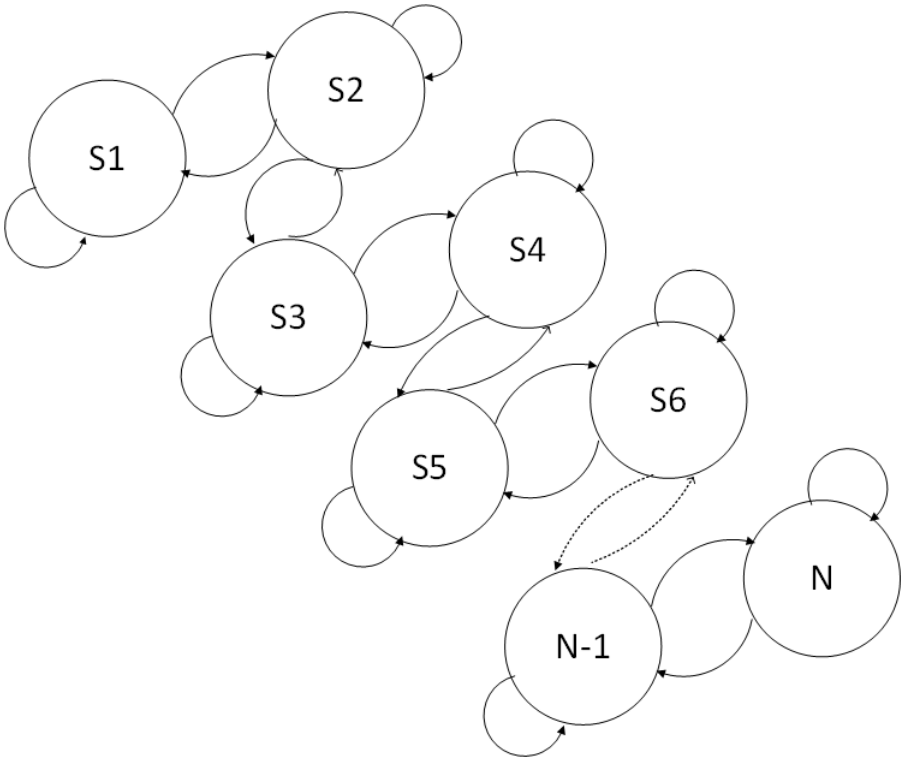


Figure 3.2: N State Markov chain

calculated by using Euclidean metric as follows [7].

$$P_d(i) = \frac{1}{N-1} \sum_{j=1, j\neq i}^{N} \sqrt{\sum_{k=1}^{D} (x_i(k) - \bar{x}_j(k))^2} \tag{3.8}$$

In Equation (3.8) N represents swarm size and D stands for dimensions of the problem.

$$E_f = \frac{P_{dg} - P_{d(min)}}{P_{d(max)} - P_{d(min)}} \in [0, 1] \tag{3.9}$$

The mean distance of all $P_d(i)$ have been derived using Equation (3.8), where the P_{dg} represents the index of global best particle and $P_{d(max)}$, $P_{d(min)}$ represents the maximum and minimum mean distance of each particle from its global best. The values derived by Equation (3.8) are then used to compute the evolutionary factor using Equation (3.9) [7].

In [7] and [8] evolutionary factor E_f has been applied and four states are derived in the $[0, 1]$ range using fuzzy classification [7]. Markov chain have been used to further improve

the search performance and convergence [8]. The idea of Markov chain is much interesting and promising. Furthermore, to improve the existing switching PSO algorithm, we have proposed an N Markovian State PSO algorithm. Initially, we have assumed that the N states concept for its applications in the real world problems. This work presents, a robust PSO algorithm by extending the switching mechanism up to N states. It further divides each of the four states to possible sub-states. The sub-states smoothly describe the unit of association for a particular state and hence the algorithm adopts a more suitable values for its parameters, where each sub-state represents the early and final stage of each state. Hence, by increasing the number of states, we assume the improved search performance in terms of the average/best evaluation results, accuracy and global convergence. Whereas, the computation burden slightly increases by increasing the number N states. An auxiliary parameter δ is used in the new velocity update Equation (3.11) in Section 3.3.2.

3.3.2 Markovian Jumping in the Velocity Update Equation

We have introduced a *mode-dependent* velocity update equation with Markovian jumping parameters having N possible states. It preserves the current state with the maximum probability and jump to the other state with minimum probability, where to improve the search performance of PSO algorithm. The process of Markovian jumping is formulated here. Markov Jumping parameter takes the value $\delta(t)(t \geq 0)$ from the set of finite states $\tau = \{1, 2, \cdots, N\}$. The probability matrix of state transitions $\Gamma = (\phi_{ij})_{N \times N}$ is stated as follows:

$$\Gamma = \begin{bmatrix} \phi_{1,1} & \phi_{1,2} & \phi_{1,3} \cdots & \phi_{1,N} \\ \phi_{2,1} & \phi_{2,2} & \phi_{2,3} \cdots & \phi_{2,N} \\ \phi_{3,1} & \phi_{3,2} & \phi_{3,3} \cdots & \phi_{3,N} \\ \vdots & \vdots & \vdots & \vdots \\ \phi_{N,1} & \phi_{N,2} & \phi_{N,3} \cdots & \phi_{N,N} \end{bmatrix}$$

$$\upsilon = \{\delta(t+1) = j \mid \delta(t) = i\} = \phi_{ij}, i, j = 1, 2, \cdots, N \tag{3.10}$$

The transition values start from i to j, and $\phi_{ij} \geq 0$, where $(i, j) \in \tau$ and $\sum_{j=1}^{N} \phi_{ij} = 1$. Subsequently, $\delta(t) = 1, 2, 3, \cdots, N$, where $(t \geq 0)$ each represents the unit of association to a particular state. The new auxiliary parameter δ has been clamped with the acceleration coefficients in the velocity update Equation (3.11) given as follows:

$$v_i(t+1) = \omega v_i(t) + c_1(\delta(t))r_1(t)(pbest_i(t) - x_i(t)) + c_2(\delta(t))r_2(t)(gbest_i(t) - x_i(t)), \quad (3.11)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (3.12)$$

whereas, an appropriate value for c_1 and c_2 have been selected according to δ. The corresponding fixed values of the acceleration coefficients will be taken according to the particle's current state. Description of N states and probability transition matrix is given bellow.

$$States = \begin{cases} 1, & 0 \leq E_f < \frac{1}{N}, \\ 2, & \frac{1}{N} \leq E_f < \frac{2}{N}, \\ 3, & \frac{2}{N} \leq E_f < \frac{3}{N}, \\ \vdots & \vdots \\ N, & \frac{N-1}{N} \leq E_f < 1 \end{cases}$$

$$\Gamma = \begin{bmatrix} \phi & 1-\phi & 0 & 0\cdots & 0 & 0 & 0 & 0 \\ \frac{1-\phi}{2} & \phi & \frac{1-\phi}{2} & 0\cdots & 0 & 0 & 0 & 0 \\ 0 & \frac{1-\phi}{2} & \phi & \frac{1-\phi}{2} & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0\cdots & 0 & \frac{1-\phi}{2} & \phi & \frac{1-\phi}{2} \\ 0 & 0 & 0 & 0\cdots & 0 & 0 & 1-\phi & \phi \end{bmatrix}$$

Basically, it is given that ϕ and $1-\phi$ are the state transition control probabilities. Iteratively, it may jump to another state or stay in the current state depending on its state transition probability. The next state can be quickly and easily predicted. The transition probability ϕ compose a significant part here in the evolutionary process. The ϕ is checked for two values 0.8 and 0.9, while 0.9 has been used throughout due to the better impact on accuracy and search diversity.

3.3.3 Computing Inertia Weight

As we have mentioned above in Section 3.2.2 that the inertia weight ω has an influential part in PSO algorithm control, global and local search performance. In this chapter, we compute the inertia weight ω by using the linearly decreasing with maximum times.

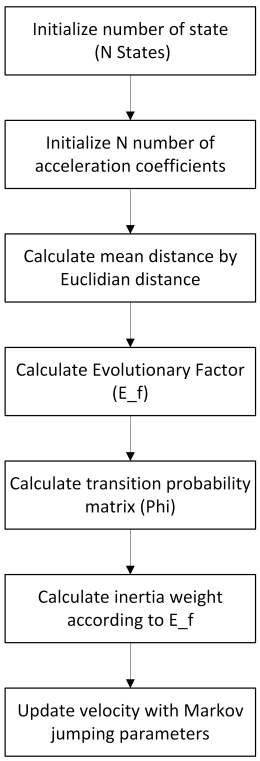


Figure 3.3: Proposed NS-MJPSO Flow Chart

$$\omega = (\omega_{max} - \omega_{min}) \times \frac{\text{iter}}{\text{maxiter}} + \omega_{min} \tag{3.13}$$

As we know that, the large values of inertia weight ω move the swarm towards the global exploration and the small inertia weight ω leads the swarm towards the local exploitation in the search space. In our experiments, we have adjusted inertia weight ω_{max} and inertia weight ω_{min} between 0.9 and 0.5 to keep the particles move faster towards the global optimum.

3.3.4 The Weights of Acceleration Coefficients

The acceleration coefficients are also based on the number of transient states. In the N number of states, N number of acceleration coefficients are manually assigned. We have assigned $c_1(\delta(0))$ and $c_2(\delta(0))$ 2.05 as the initial value here and further the algorithm adopts the adjusted values from the array according to the current evolutionary state. The strategy for selecting the appropriate value of the accelerating coefficients is determined as follows.

1. The first one is the jumping-out state, which is further categorised according the total number of states N and the appropriate value of accelerating coefficient is assigned in time t with the unit of being in the jumping-out state. For instance, whether it needs a high jump from its current state or a small jump. Therefore, large values are assigned to c_2 and small values are assigned to c_1 to let the swarm fly towards the global optimum. We assume the maximum values of c_2 as 2.2 and 2.1 and c_1 are assigned 1.8 and 1.7, if N is 8 and can adjusted further if N is otherwise.

2. The next is determined as exploration state, in this state, it is very important to explore the entire search space and converge quickly. For that reason c_1 has assigned a large value 2.2 and 2.1 and a small value 1.8 and 1.7 to c_2.

3. The third one is exploitation state, in this state we assume the swarm nearly converged to the global optimum or we can say that it is near to the convergent state. Therefore, we consider a large value 2.1 of c_1 and a small value 1.9 of c_2.

4. The last one is the convergence state, in this state the particles group together on a single global optimal position. Thus, to avoid premature convergence and balance the search between local and global regions on the search space, we assign the same moderate value 2 to both c_1 and c_2.

In contrast to other methods [7] of calculating accelerating coefficients our approach is simpler and effective. The current state is determined from the probability matrix [8]. It has been observed that the current state has the large value of probability to remain in the same state and has the small probability to jump to the other state. This ensures the very fast convergence and avoid the swarm to fly towards the local optimum. The Markovian jumping process is demonstrated by the following Pseudocode 2.

3.4 Experimental Work

The proposed NS-MJPSO has been evaluated 12 commonly used benchmark functions that are given in Table 3.1 [14]. Initially, the proposed NS-MJPSO is applied to the 12 functions f_1 to f_{12} in 30 dimensions. Then the evaluation results are compared with published values of six state-of-the-art algorithms. The reported results are derived from the 30 independent trials. All the work have been conducted on a PC with an Intel Core i5-3320M 2.6 GHz CPU and Microsoft Windows 10 Pro 64-bit system. The benchmark test experiments of the 12 uni-modal and multi-modal problems for the proposed NS-MJPSO and other PSO variants in comparison are all coded in Matlab R2015a. It is also worth to be mentioned

Algorithm 2 Markov Jumping Process

$N \leftarrow$ Number of states
$D \leftarrow$ Dimension
$S \leftarrow$ Population
$X \leftarrow$ Current Position
$C \leftarrow$ Acceleration Coefficients
for Each particle i **do**
    Calculate the mean distance using Equation (3.8)
    $temp_2 \leftarrow 0$
    for k **do** $\leftarrow 1 : S$
        temp $\leftarrow 0$
        for j **do** $\leftarrow 1 : D$
            temp $\leftarrow (X(i,j) - X(k,j))^2 + temp$
        end for
        $temp_2 \leftarrow temp_2 + \sqrt{temp}$
    end for
    $Dis_i \leftarrow temp_2/S$
end for
calculate E_f and current state
$E_f \leftarrow (Dis_j - min(Dis))/(max(Dis) - min(Dis))$
for **do**$i \leftarrow 0$:N States
    if **then**i/N States $\leq E_f < (i+1)$/N States
        $\delta \leftarrow$ current state i
    end if
end for
predict the next state using transition matrix Γ
$\phi(1,1) \leftarrow 0.9$ initially fixed
$\phi(N_s, N_s) \leftarrow 0.1$ minimum
for Each State i **do** $\leftarrow 0 : N_s$
    calculate the next state using the ϕ given values
end for

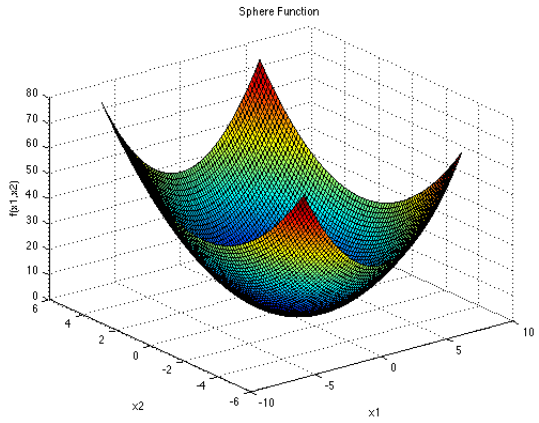
that because of the N state Markov jumping mechanism and the evolutionary control of the current states, the algorithm converge to its optimum in the early stages of their function evaluations.

Table 3.1: The Benchmark Functions

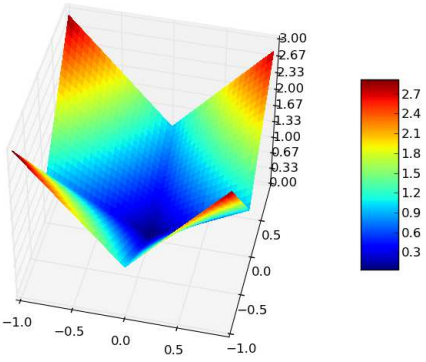
Name	Function	Dimension	Search space
Sphere (f_1)	$f(x_1 \cdots x_n) = \sum_{i=1}^{n} x_i^2$	30	[-100, 100]
Schwefel 2.22 (f_2)	$f(x_0 \cdots x_n) = \sum_{i=0}^{n} \lvert x_i \rvert + \prod_{i=0}^{n} \lvert x_i \rvert$	30	[-10, 10]
Schwefel 1.2 (f_3)	$f(x_1 \cdots x_n) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	30	[-100, 100]
Schwefel 2.21 (f_4)	$f(x_0 \cdots x_n) = max_i\{\lvert x_i \rvert, 1 \leq i \leq n\}$	30	[-100, 100]
Rosenbrock (f_5)	$f(x_1 \cdots x_n) = \sum_{i=1}^{n-1} (100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2)$	30	[-30, 30]
Step (f_6)	$f(x_1 \cdots x_n) = \sum_{i=1}^{n} x_i^2$	30	[-100, 100]
Schwefel (f_7)	$f(x_1 \cdots x_n) = 418.982887.n + \sum_{i=1}^{n} (-x_i.sin(\sqrt{\lvert x_i \rvert}))$	30	[-500, 500]
Rastrigin (f_8)	$f(x_1 \cdots x_n) = 10n + \sum_{i=1}^{n} (x_i^2 - 10cos(2\pi x_i))$	30	[-5.12, 5.12]
Ackley's (f_9)	$f(x_0 \cdots x_n) = -20exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - exp(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)) + 20 + e$	30	[-32, 32]
Griewank (f_{10})	$f(x_1 \cdots x_n) = 1 + \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}})$	30	[-600, 600]
Penalized 1 (f_{11})	$f(x_1 \cdots x_n) = \frac{\pi}{n}\{10sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10sin^2(\pi y_{i+1})] + (y_n - 1)^2\}$ $+ \sum_{i=1}^{n} u(x_i, 10, 100, 4)$, $y = 1 + (\frac{1}{4})(x_i + 1)$	30	[-50, 50]
Penalized 2 (f_{12})	$f(x_1 \cdots x_n) = 0.1\{sin^2(3\pi x_1) + \sum_{i=1}^{n-1}(x_n - 1)^2[1 + sin^2(\pi x_{i+1})] + (x_n - 1)^2\}$ $+ \sum_{i=1}^{n} u(x_i, 5, 100, 4)$,	30	[-50, 50]

3.4.1 Performance of the Proposed NS-MJPSO on Benchmark Functions

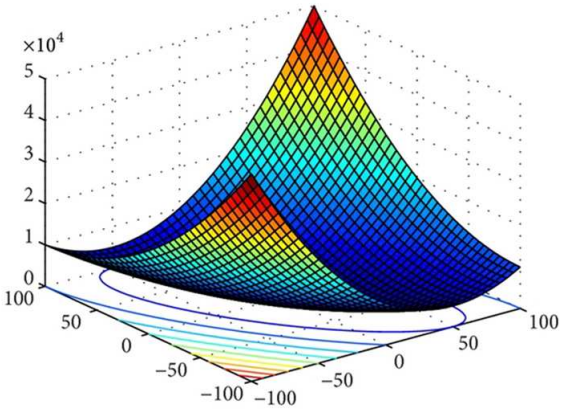
In the experimental analysis of the 12 benchmark functions, we have selected the six illustrative PSO modified versions for comparing the performance of proposed NS-MJPSO algorithm on the same problems. First one is the recently developed PSO variant named as SL-PSO [14], second is the local-neighbourhood PSO (Local-PSO) [58], third is the global best version (GPSO) [46], fourth is the dynamic multi-swarm version of PSO (DMS-PSO) [60; 62], fifth one is the fully-informed PSO (FIPS) [59], the sixth and last one is comprehensive-learning PSO (CLPSO) [4]. The required parameters along-with the values are described here in Table 3.2. The proposed NS-MJPSO has presented outstanding performance on 10 out 12 problems (f_1 to f_6 and f_8 to f_{12}), containing uni-modal and multi-modal problems. We have shown the performance of proposed NS-MJPSO algorithm individually for each function in Table 3.3 to Table 3.14 and Figure 3.5 to Figure 3.16 as follows:



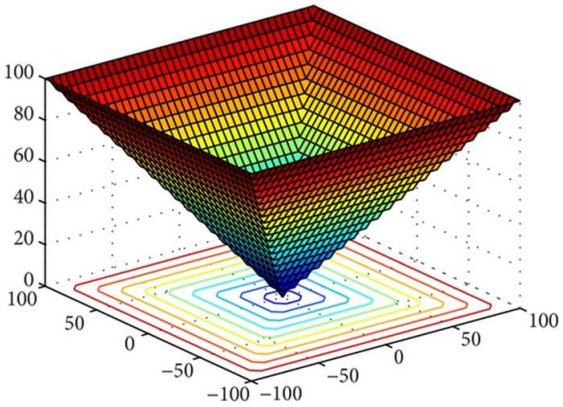
(f_1) The Sphere Function.



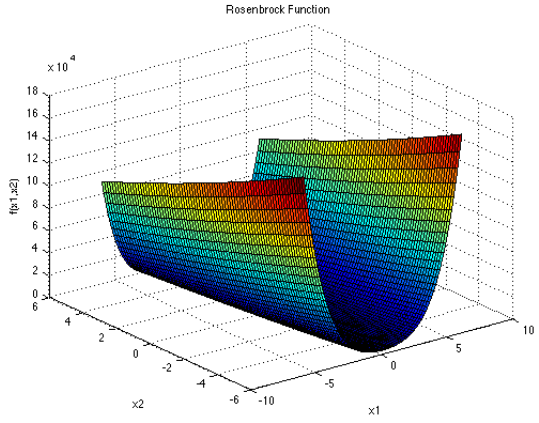
(f_2) The Schwefel 2.22 Function



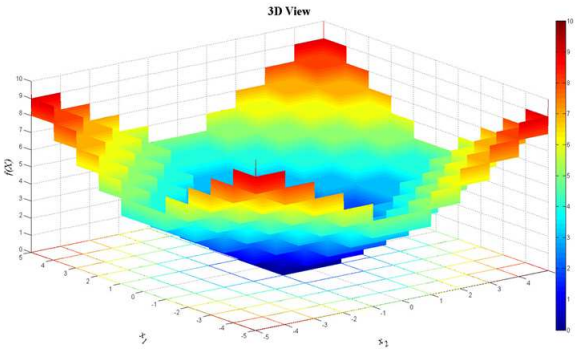
(f_3) The Schwefel 1.2 Function.



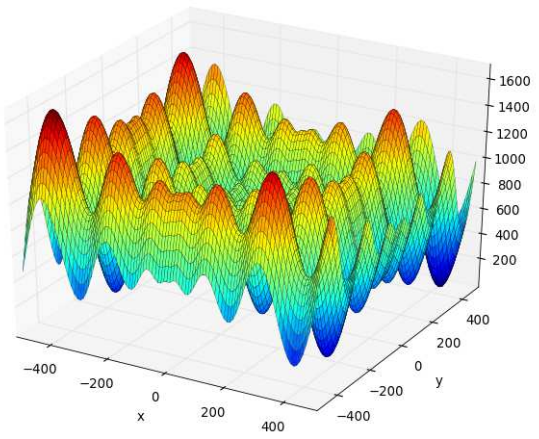
(f_4) The Schwefel 2.21 Function



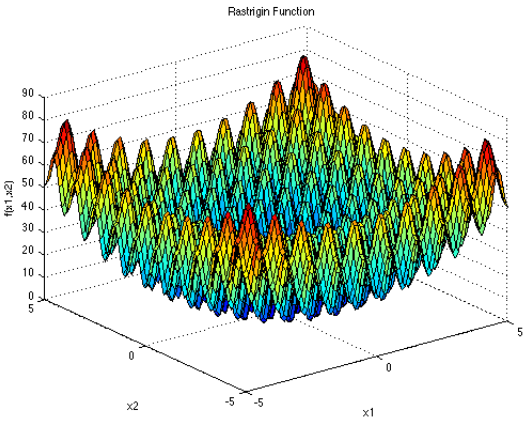
(f_5) The Rosenbrock Function.



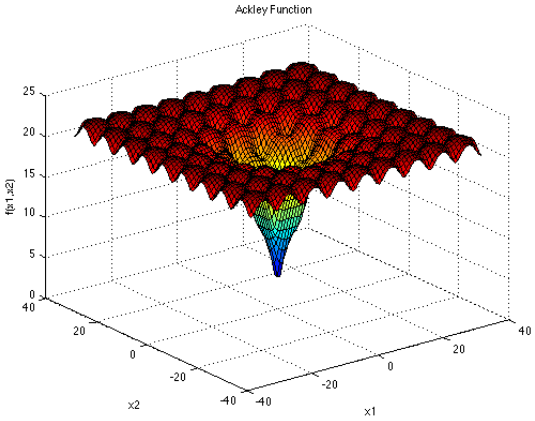
(f_6) The Step Function



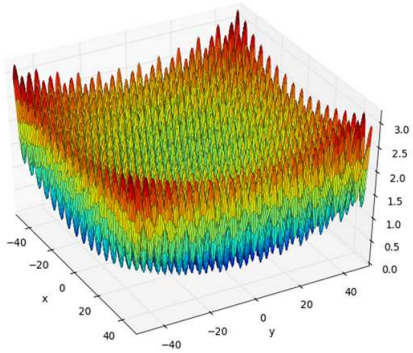
(f_7) The Schwefel Function.



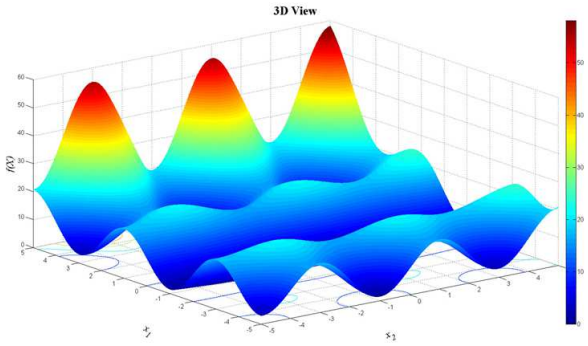
(f_8) The Rastrigin Function



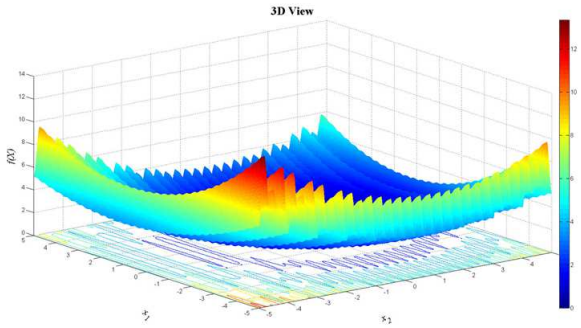
(f_9) The Ackley's Function.



(f_{10}) The Griewank Function



(f_{11}) The Penalized 1 Function.



(f_{12}) The Penalized 2 Function

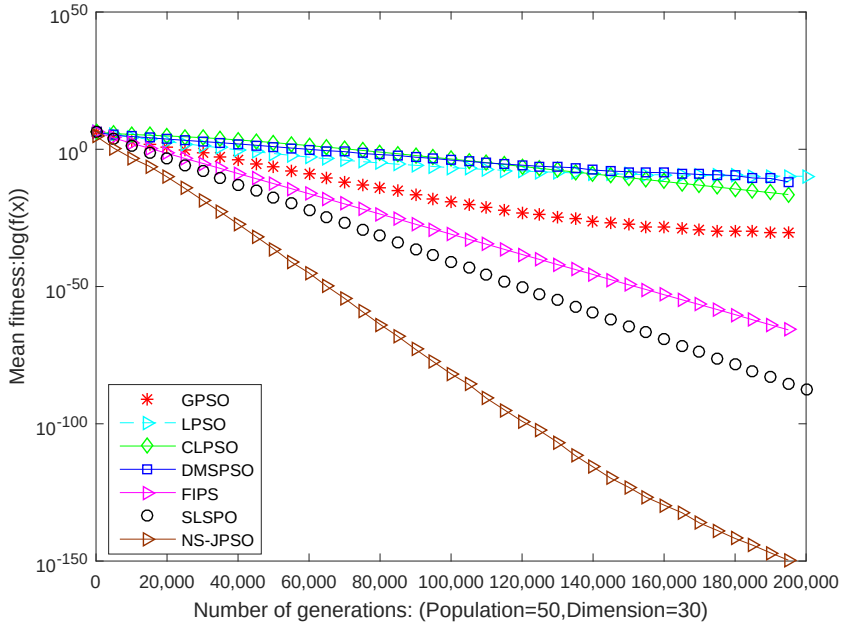
Figure 3.4: The 3D representation of f_{11} and f_{12}

Table 3.2: Parameter coefficients of the PSO variants for comparison

Algorithm	Inertia weight	Acceleration Coefficients
SL-PSO	automatic	automatic
LPSO	$[0.9, 0.4]$	$c_1 = c_2 = 2.0$
GPSO	$[0.9, 0.4]$	$c_1 = c_2 = 2.0$
DMS-PSO	0.729	$c_1 = c_2 = 1.49445, m = 3, R = 15$
FIPS	$\chi = 0.729$	$c_1 + c_2 = 4.1$
CLPSO	$[0.9, 0.7]$	$c_1 = c_2 = 1.49445$
NS-MJPSO	$[0.9, 0.5]$	$c_1 = [2, N], c_2 = [2, N], \phi = 0.9, N$

Table 3.3: Simulation results for Sphere function

Algorithm	Mean	Best Value	Std Dev
NS-MJPSO	2.16E-150	2.79E-161	1.82E-144
SL-PSO	4.24E-90	5.08E-92	3.26E-89
LPSO	4.89E-12	1.80E-14	4.86E-12
FIPS	7.23E-70	4.78E-71	6.55E-70
DMS-PSO	3.81E-15	4.97E-20	1.02E-14
CLPSO	6.32E-19	1.69E-19	4.56E-19
GPSO	5.56E-33	3.30E-45	1.93E-32



(f_1) The Sphere Function

Figure 3.5: The mean fitness of f_1 30 dimensional problems

NS-MJPSO The new algorithm NS-MJPSO has produced best results for f_1 Sphere function in terms of the minimum error, mean and standard deviation. The robustness and accuracy of the NS-MJPSO is shown by the mean value achieved in Table 3.3.

SL-PSO, FIPS The SL-PSO and FIPSO have the similar good performance in second and third ranking.

GPSO has also produced the best performance for f_1 Sphere function in terms of the best minimum value.

LPSO has resulted poor performance for f_1 Sphere function in terms of its statistical evaluations.

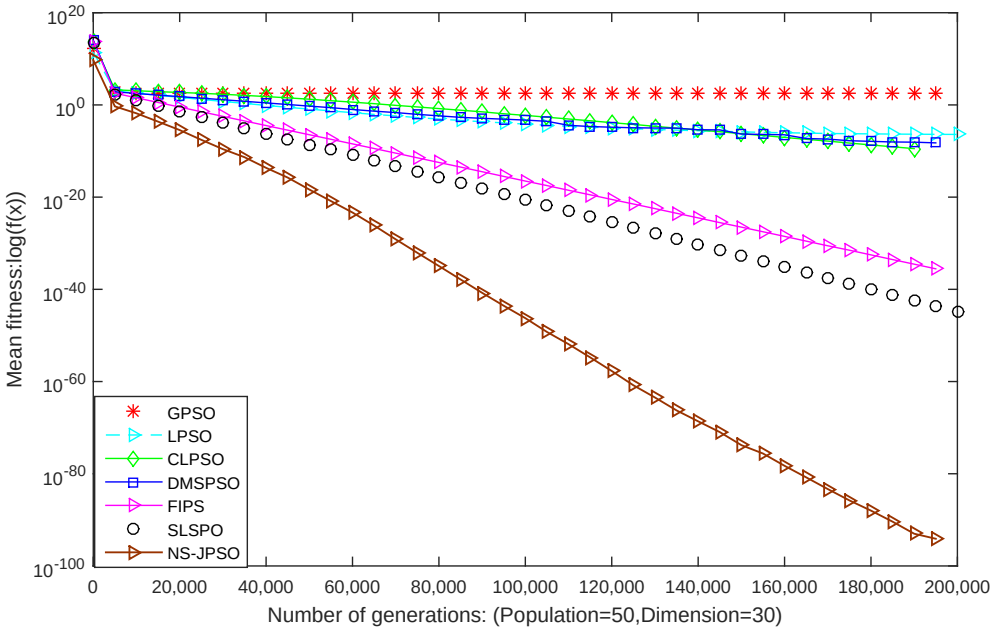
CL-PSO has also relatively poor results for same function.

DMS-PSO This algorithm have the similar performance as CL-PSO for f_1

NS-MJSPSO The new algorithm NS-MJPSO has produced best results for f_2 Schwefel 2.22 function in terms of the minimum error, mean and standard deviation. The results produced by NS-MJPSO are more robust and accurate as given in Table 3.4.

Table 3.4: The Schwefel 2.22

Algorithm	Mean	Best Value	Std Dev
NS-MJPSO	7.93E-95	2.51E-98	6.40E+00
SL-PSO	1.50E-46	1.09E-47	6.27E-47
LPSO	1.33E-08	9.36E-10	1.39E-08
FIPS	9.99E-39	2.71E-39	5.40E-39
DMS-PSO	3.29E-11	1.42E-14	8.70E-11
CLPSO	7.49E-12	4.70E-12	2.28E-12
GPSO	9.67E+00	1.85E-28	1.03E+01



(f_2) The Schwefel 2.22 Function

Figure 3.6: The mean fitness of f_2 on 30 dimensional problems

SL-PSO, FIPS The SL-PSO and FIPSO have again the similar good performance for f_2 and stand on rank 2 and 3 respectively.

GPSO has also produced the best performance for f_2 Schwefel 2.22 function in terms of the best minimum value.

LPSO has resulted poor performance for f_2 Schwefel 2.22 function in terms of its statistical evaluations.

CL-PSO has also again produced poor results for f_2 as well.

DMS-PSO This algorithm have the similar performance as CL-PSO for f_2

Table 3.5: Function Schwefel 1.2

Algorithm	Mean	Best Value	Std Dev
NS-MJPSO	1.48E-23	5.45E-27	9.13E+02
SL-PSO	4.66E-07	5.95E-08	9.69E-07
LPSO	2.75E+01	8.10E+00	1.43E+01
FIPS	1.16E+00	3.58E-01	6.05E-01
DMS-PSO	8.35E+01	1.06E+01	5.51E+01
CLPSO	1.06E+03	6.74E+02	3.20E+02
GPSO	2.22E+03	4.44E-05	3.46E+03

NS-MJPSO Our algorithm NS-MJPSO is on the top for f_3 Schwefel 1.2 function in terms of the best minimum value, mean and standard deviation. In addition to that NS-MJPSO is more robust and with accurate results for f_3 given in Table 3.5.

SL-PSO The SL-PSO has relatively good performance for f_3.

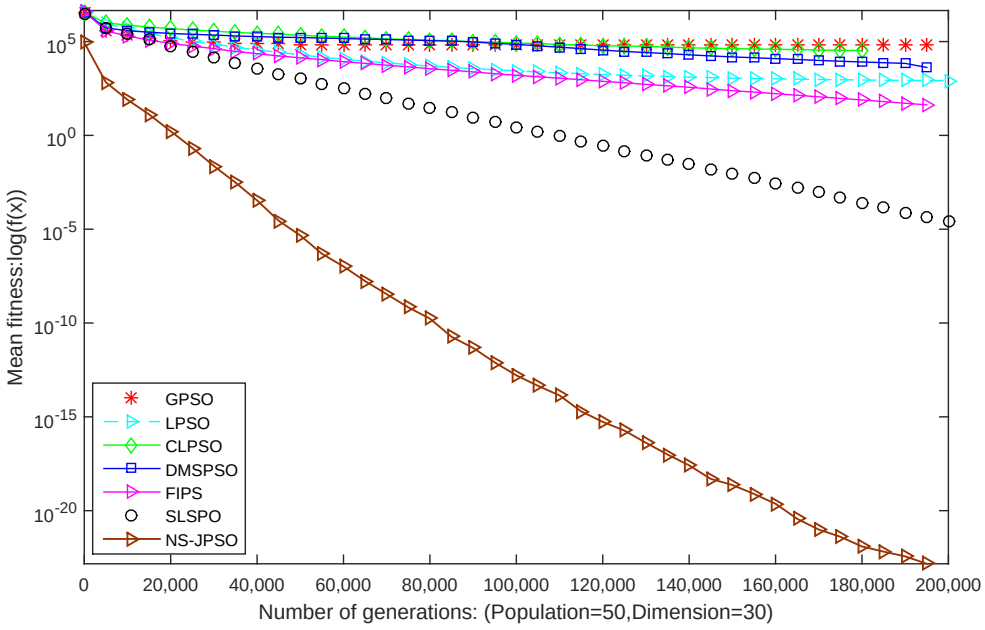
GPSO has also produced the best performance for f_3 Schwefel 1.2 function in terms of the best minimum value.

FIPS has poor performance for f_3.

LPSO has resulted poor performance for f_3 Schwefel 1.2 function.

CL-PSO has also again produced poor results for f_3 as well.

DMS-PSO This algorithm have the similar performance as CL-PSO for f_3

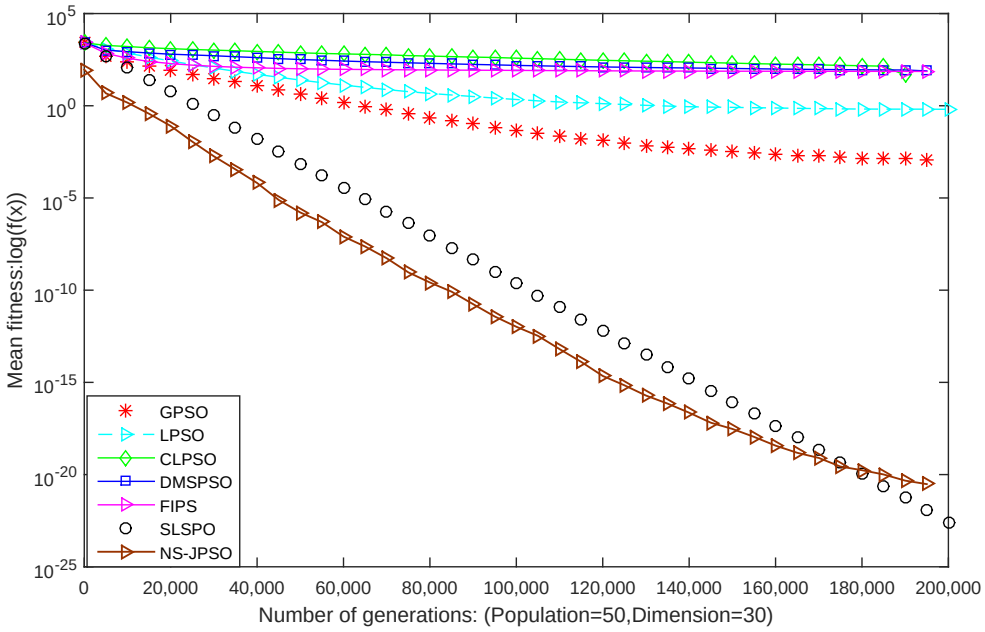


(f_3) The Schwefel 1.2 Function

Figure 3.7: The mean fitness of f_3 30 dimensional problems

Table 3.6: Function Schwefel 2.21

Algorithm	Mean	Best Value	Std Dev
NS-MJPSO	3.04E-21	1.48E-23	8.22E-21
SL-PSO	1.17E-24	1.44E-25	5.62E-25
LPSO	2.14E-02	8.23E-03	8.04E-03
FIPS	2.42E+00	3.60E-01	1.15E+00
DMS-PSO	2.14E+00	8.52E-01	8.80E-01
CLPSO	4.50E+00	3.32E+00	5.15E-01
GPSO	3.87E-05	3.71E-06	3.27E-05



(f_4) The Schwefel 2.1 Function

Figure 3.8: The mean fitness of f_4 on 30 dimensional problems

SL-PSO The SL-PSO has produced the best performance for f_4 in terms of the best minimum value, mean and standard deviation.

NS-MJPSO The newly developed NS-MJPSO has second better performance in terms of accuracy for f_4 Schwefel 2.21 function, where in computation time is better than all the algorithms used in comparison here.

GPSO has also produced the best performance for f_4 Schwefel 2.21 function in terms of the best minimum value.

FIPS has poor performance for f_4.

LPSO has resulted poor performance for f_4 Schwefel 2.21 function.

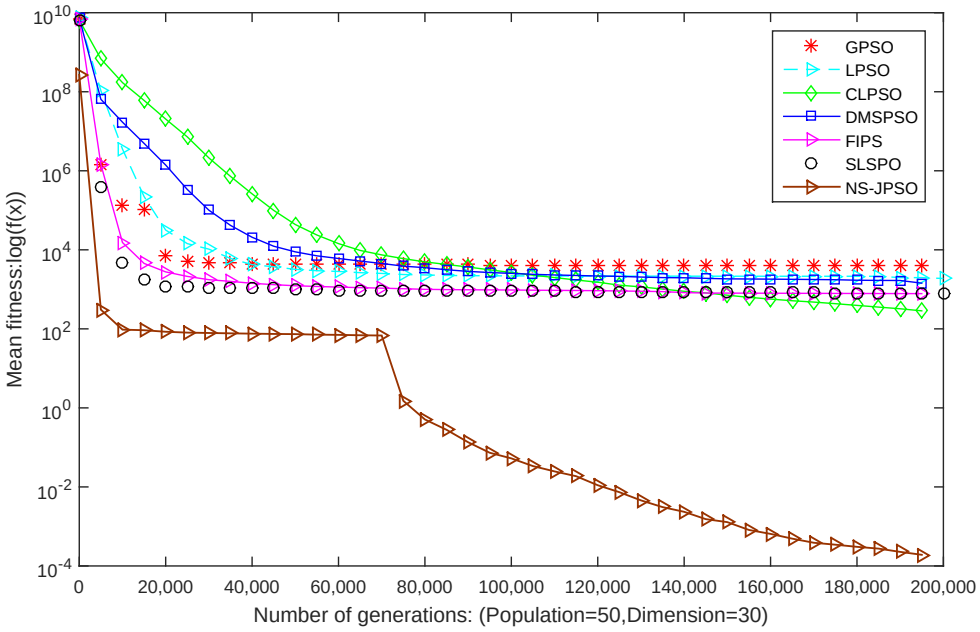
CL-PSO has also again produced poor results for f_4 as well.

DMS-PSO This algorithm have the similar performance as CL-PSO for f_4

NS-MJPSO The newly developed NS-MJPSO stands number one for f_5 Rosenbrock function in terms of robustness and accuracy shown in Table 3.7.

Table 3.7: The Function Rosenbrock

Algorithm	Mean	Best Value	Std Dev
NS-MJPSO	1.92E-04	8.47E-09	1.64E+04
SL-PSO	2.15E+01	9.79E+00	2.23E+01
LPSO	6.27E+01	7.32E+00	5.98E+01
FIPS	2.59E+01	1.25E-01	1.71E+01
DMS-PSO	3.86E+01	2.74E-01	3.03E+01
CLPSO	9.55E+00	1.73E+00	7.73E+00
GPSO	1.31E+02	3.84E-01	5.59E+02



(f_5) The Rosenbrock Function

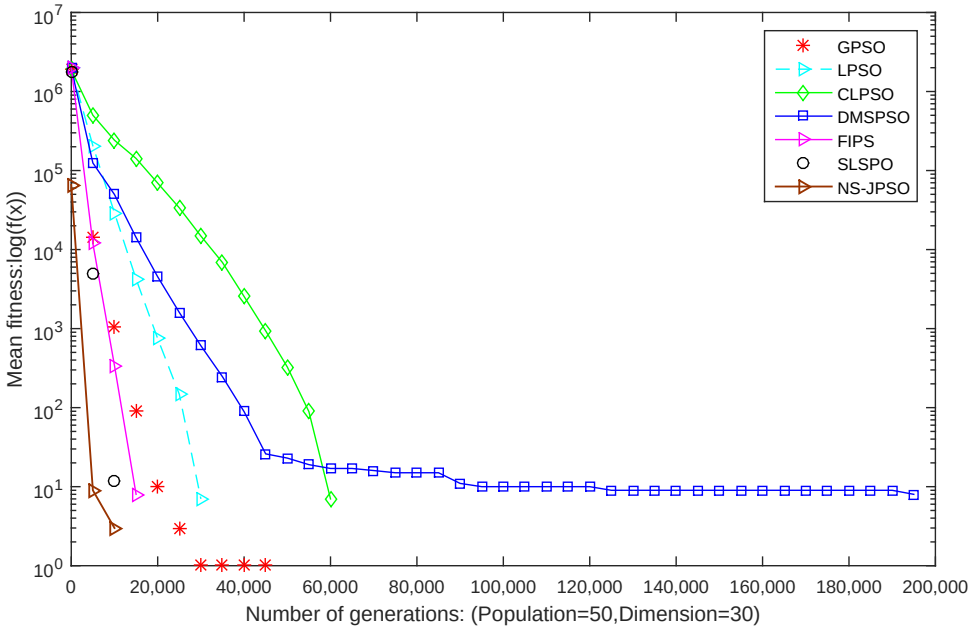
Figure 3.9: The mean fitness of f_5 30 dimensional problems

SL-PSO The SL-PSO, CL-PSO and LPSO have similar average performance for f_5.

GPSO , FIPS and DMS-PSO have similar and relatively better performance for f_5 Rosenbrock function in terms of the best minimum value.

Table 3.8: The Step Function

Algorithm	Mean	Best Value	Std Dev
NS-MJPSO	0.00E+00	0.00E+00	0.00E+00
SL-PSO	0.00E+00	0.00E+00	0.00E+00
LPSO	0.00E+00	0.00E+00	0.00E+00
FIPS	0.00E+00	0.00E+00	0.00E+00
DMS-PSO	2.67E-01	0.00E+00	5.21E-01
CLPSO	0.00E+00	0.00E+00	0.00E+00
GPSO	0.00E+00	0.00E+00	0.00E+00



(f_6) The Step Function

Figure 3.10: The mean fitness of f_6 on 30 dimensional problems

NS-MJPSO In f_6 all the algorithms have shown similar performance. NS-MJPSO has the robust adaptability for such type of problems, where it convergence to global optimum in shorter computation.

GPSO has also produced the best performance for f_6 Step function.

FIPS has also produced the best performance for f_6 Step function.

LPSO has also produced the best performance for f_6 Step function.

CL-PSO has also produced the best performance for f_6 Step function.

DMS-PSO has also produced the best performance for f_6 Step function. However, slow performance and some inconsistency is observed.

Table 3.9: The Schwefel Function

Algorithm	Mean	Best Value	Std Dev
NS-MJPSO	2.99E+03	1.40E+03	2.16E+03
SL-PSO	1.56E+03	8.29E+02	4.24E+02
LPSO	1.87E+03	9.51E+02	5.30E+02
FIPS	2.70E+03	1.34E+03	7.56E+02
DMS-PSO	-2.55E-01	-7.66E+00	1.40E+00
CLPSO	4.85E-13	0.00E+00	8.18E-13
GPSO	5.52E+03	2.82E+03	2.33E+03

NS-MJSPSO In f_7 Schwefel function all the algorithms including NS-MJPSO, have the similar poor performance in terms of best values. However, NS-MJPSO is faster than the other algorithms in comparison.

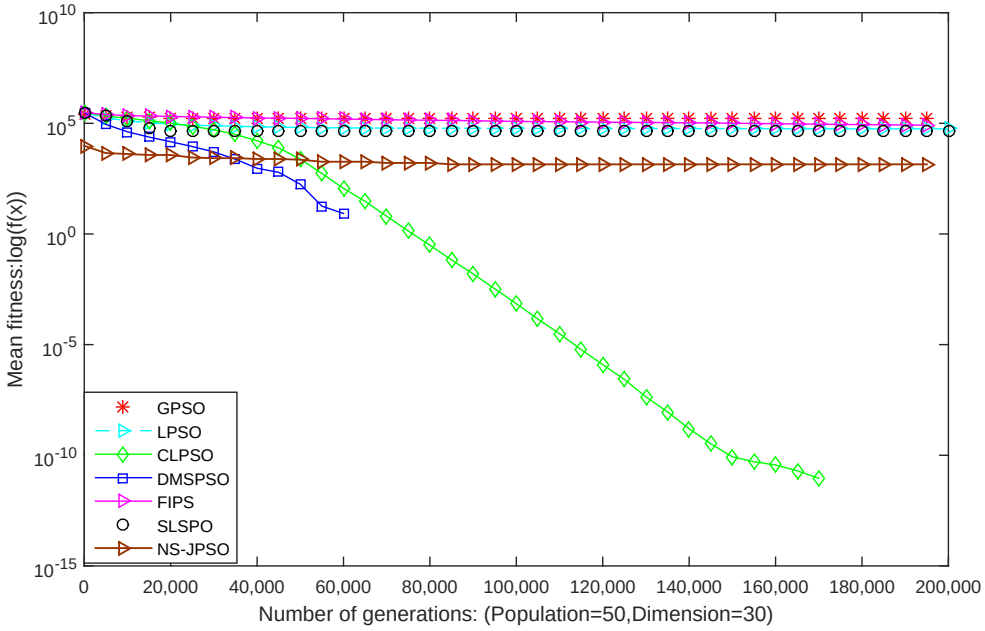
CL-PSO has the best performance for f_7 Schwefel function Figure 3.11.

SL-PSO has also similar performance as NS-MJPSO.

GPSO , FIPS and LPSO has same performance for f_7

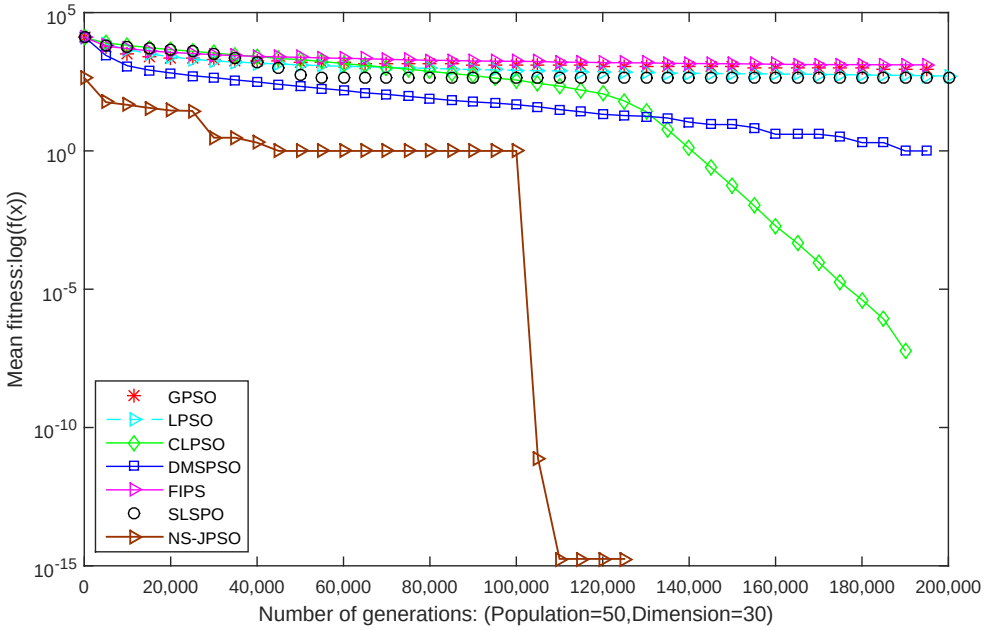
Table 3.10: Rastrigin Function

Algorithm	Mean	Best Value	Std Dev
NS-MJPSO	0.00E+00	0.00E+00	0.00E+00
SL-PSO	1.55E+01	6.96E+00	4.16E+00
LPSO	1.69E+01	4.01E+00	1.05E+01
FIPS	4.25E+01	2.70E+01	6.66E+00
DMS-PSO	3.32E-02	1.78E-15	1.82E-01
CLPSO	6.13E-09	4.23E-10	6.86E-09
GPSO	3.05E+01	1.98E-07	3.15E+01



(f_7) The Schwefel Function

Figure 3.11: The mean fitness of f_7 30 dimensional problems



(f_8) The Rastrigin Function

Figure 3.12: The mean fitness of f_8 on 30 dimensional problems

NS-MJSPSO In f_8 Rastrigin function NS-MJPSO has the performance in comparison to all algorithms in terms of best evaluation values and also the computation time.

CL-PSO has the second best performance for f_8 in comparison to NS-MJPSO.

SL-PSO , and LPSO have the same performance. However, the SL-PSO is much faster than LPSO.

GPSO , has also good performance in terms of best value.

DMS-PSO has also good performance, where as usual has much computation time in comparison Figure 3.12.

Table 3.11: The Ackley's Function

Algorithm	Mean	Best Value	Std Dev
NS-MJPSO	1.33E-14	6.22E-15	3.41E-15
SL-PSO	5.54E-15	2.66E-15	1.23E-15
LPSO	2.87E-06	7.68E-08	5.48E-06
FIPS	7.16E-15	6.22E-15	2.46E-15
DMS-PSO	1.49E-08	6.84E-11	3.71E-08
CLPSO	2.98E-10	1.24E-10	9.33E-11
GPSO	9.25E-01	6.22E-15	3.52E+00

NS-MJSPSO In f_9 Ackley's function, NS-MJPSO, FIPS, and GPSO have the same minimum best values. However, the proposed NS-MJPSO is much faster than others.

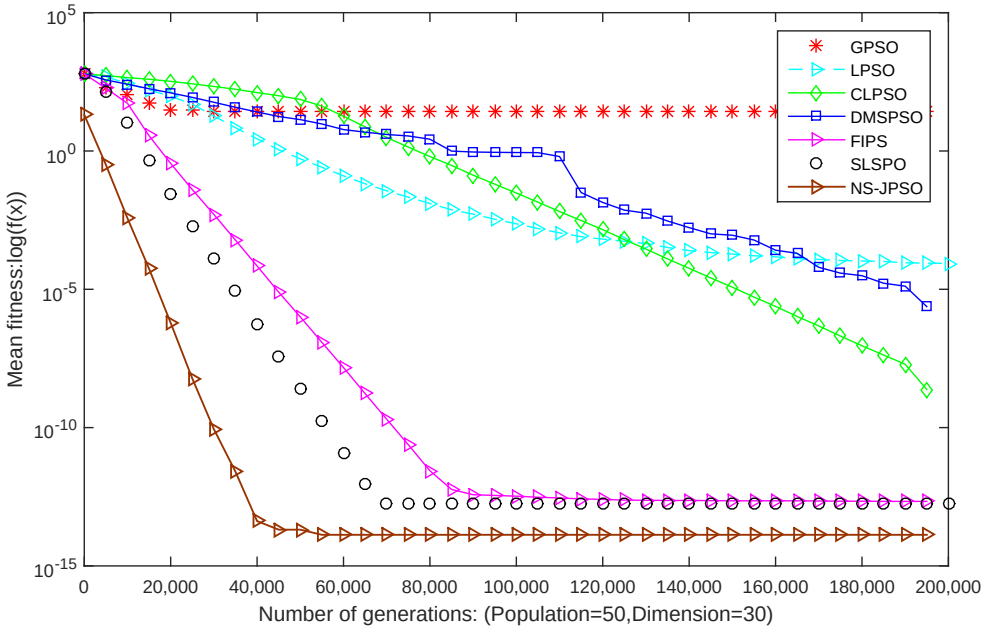
SL-PSO has the best performance f_9 Ackley's function, and has the minimum value.

DMS-PSO , and CLPSO have the similar average performance for f_9 Ackley's function Figure 3.13.

NS-MJSPSO In f_{10} Griewank function, NS-MJPSO, SL-PSO, FIPS, DMS-PSO, and GPSO have the same minimum values. As shown in Figure 3.14 NS-MJPSO has converged very early.

CL-PSO , and LPSO have the same performance for f_{10} Griewank function Figure 3.14.

NS-MJSPSO In f_{11} Penalized 1 function, NS-MJPSO, has converged to the global optimum very quick. SL-PSO, FIPS, and GPSO have the same best values. However, the GPSO and FIPS are relatively very slower than NS-MJPSO and SLPSO.



(f_9) The Ackley's Function

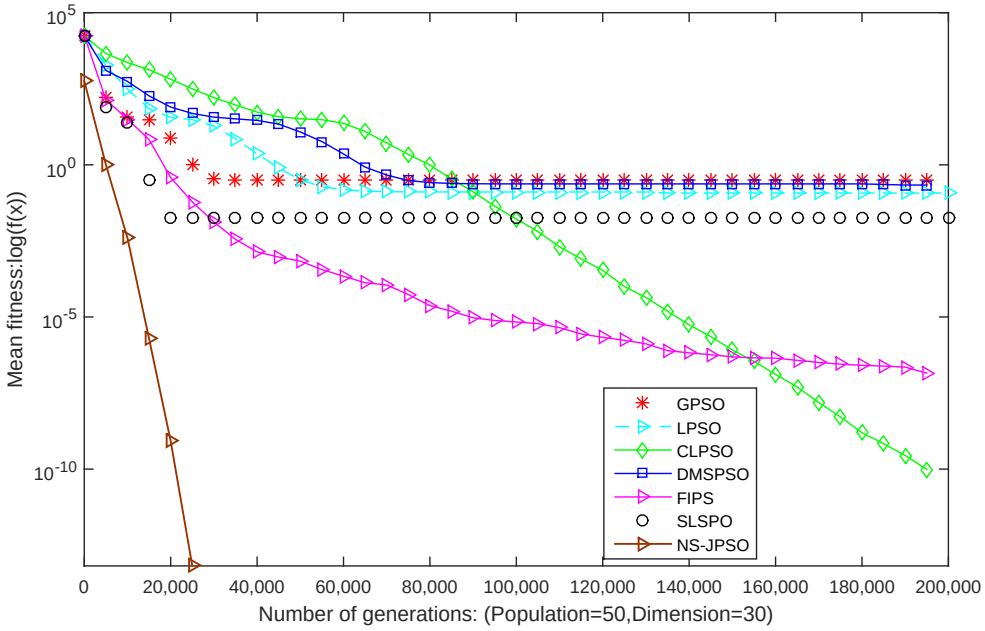
Figure 3.13: The mean fitness of f_9 30 dimensional problems

Table 3.12: The Griewank Function

Algorithm	Mean	Best Value	Std Dev
NS-MJPSO	0.00E+00	0.00E+00	0.00E+00
SL-PSO	0.00E+00	0.00E+00	0.00E+00
LPSO	3.94E-03	2.44E-13	6.29E-03
FIPS	4.48E-09	0.00E+00	2.34E-08
DMS-PSO	7.22E-03	0.00E+00	1.06E-02
CLPSO	2.20E-12	7.22E-15	4.28E-12
GPSO	1.05E-02	0.00E+00	1.44E-02

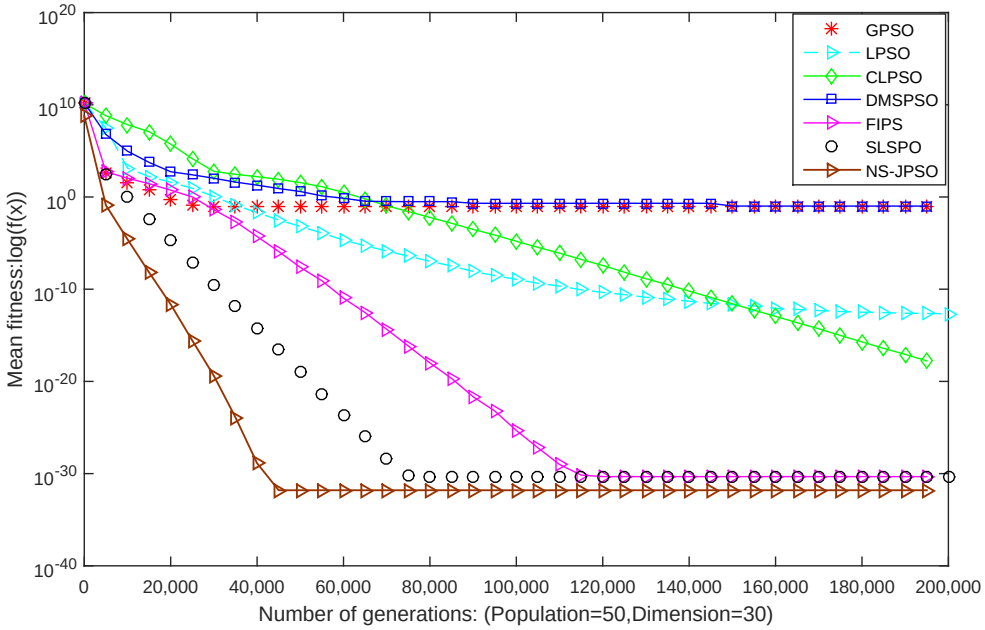
Table 3.13: The Penalized 1 Function

Algorithm	Mean	Best Value	Std Dev
NS-MJPSO	1.57E-32	1.57E-32	5.76E-02
SL-PSO	1.57E-32	1.57E-32	5.57E-48
LPSO	6.97E-15	2.45E-16	1.20E-14
FIPS	1.57E-32	1.57E-32	5.57E-48
DMS-PSO	3.46E-03	1.24E-21	1.89E-02
CLPSO	3.09E-20	9.08E-21	1.56E-20
GPSO	3.46E-03	1.57E-32	1.89E-02



(f_{10}) The Griewank Function

Figure 3.14: The mean fitness of f_{10} on 30 dimensional problems



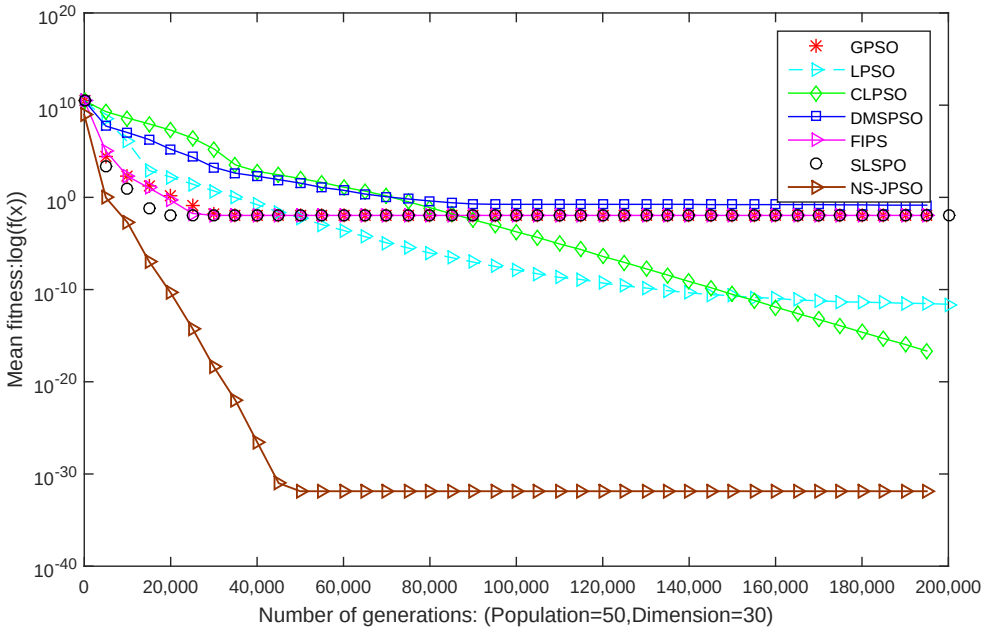
(f_{11}) The Penalized 1 Function

Figure 3.15: The mean fitness of f_{11} 30 dimensional problems

DMS-PSO , and CL-PSO have the same performance in terms of best values for f_{11} Penalized 1 function Figure 3.15

Table 3.14: The Penalized 2 Function

Algorithm	Mean	Best Value	Std Dev
NS-MJPSO	1.35E-32	1.35E-32	3.35E-03
SL-PSO	1.35E-32	1.35E-32	2.01E-03
LPSO	8.45E-14	3.36E-15	1.27E-13
FIPS	3.66E-04	1.35E-32	2.01E-03
DMS-PSO	4.76E-03	1.47E-18	8.98E-03
CLPSO	4.17E-19	1.09E-19	2.82E-19
GPSO	1.43E-16	1.35E-32	7.82E-16



(f_{12}) The Penalized 2 Function

Figure 3.16: The mean fitness of f_{11} 30 dimensional problems

NS-MJSPSO In f_{12} Penalized 2 function, NS-MJPSO, has converged to the global optimum very quick. SL-PSO, FIPS, and GPSO have the same best values. However, the GPSO and FIPS are relatively very slower than NS-MJPSO and SLPSO.

DMS-PSO , and CL-PSO have again the same performance in terms of best values for f_{12} Penalized 2 function Figure 3.16

3.4.2 Computation Time of the Proposed NS-MJPSO

The mean time for all the algorithms in shown in Figure 3.17. The Local PSO (LPSO) has the smaller average computation time. However, our newly developed has the second level faster computation time along-with having the best minimum values for most of the uni-modal and multi-modal problems.

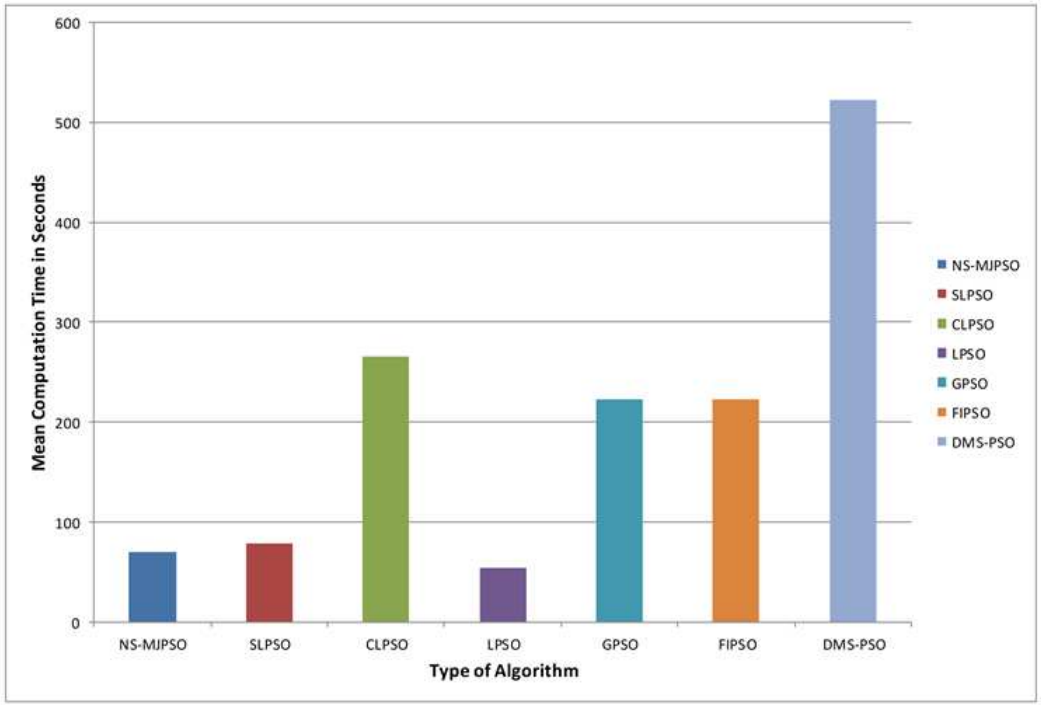


Figure 3.17: Average computation of the proposed NS-MJPSO in comparison

3.5 Summary

This chapter has contributed a new stochastic based PSO algorithm named N State Markov Jumping Particle Swarm Optimization (NS-MJPSO) algorithm. NS-MJPSo has combined the term evolutionary method for population distribution and Markov jumping for stochastic analysis of the search space. To further illustrate the performance of this newly developed NS-MJPSO algorithm, we have applied to evaluate 12 uni-modal, multi-modal and non-linear benchmark functions. Furthermore, we have re-implemented Social-learning PSO (SL-PSO), Globalbest PSO (GPSO), Localbest PSO(LPSO), Fully-informed PSO(FIPS), Dynamic Multi-Swarm PSO (DMS-PSO) and Comprehensive-learning PSO (CL-PSO). The performance of all the above mentioned algorithms is evaluated on the

same functions. It has been clearly visualized in graphs and tables that the novel NS-MJPSO has the significant performance in terms of solution quality with increasing the N states. The algorithm has performed well for most of the mathematical benchmark functions in the trials. The computational burden is slightly increased, but the sophisticated technique of Markovian Jumping has significantly contributed to a robust method for optimization. The proposed NS-MJPSO can be used for any kind of optimization problems, where accuracy is the main concern for the problem.

Chapter 4

A Novel N State Switching PSO Algorithm

4.1 Introduction

In this chapter we introduce a novel N State Switching Particle Swarm Optimization (NS-SPSO) algorithm. The NS-SPSO algorithm is based on the assumptions to further improve the performance of NS-MJPSO by excluding some complex steps. The first step is to determine the jumping probability according to the domain knowledge. Practically, it is very difficult to have enough domain knowledge about the optimization problem. The second step is to reduce the computational complexity and burden induced by Markov chain. In the novel NS-SPSO algorithm, the velocity is updated purely according to the evolutionary factor value. The particle switches from one state to another state according to the assessment of its current evolutionary factor. Furthermore, the choice of staying in the current state or switching to the other state is made by how large the value of evolutionary factor is. First of all the population distribution and the mean distance by using Euclidean distance are determined. The evolutionary factor is then derived by using the population distribution and mean distance of each particle from the global best. In the population distribution, we determine that how far is the particle away from each other and also its global optimum. The main states are described as exploration, exploitation, convergence and jumping-out states. However, in the N states, we further divide the main states into sub-states or stages. Each state is assigned a value in the range of $(0, 1)$.

The parameters are assigned adequate weights according to each sub-state or stage. In the N states, N number of acceleration coefficients are assigned, but the appropriate value is taken during the evaluation process according to the current state. Subsequently, the algorithm converges to the optimum in few iterations. For that reason, we have adopted the linearly time decreasing inertia weight concept for NS-SPSO algorithm. Extensive simulations have been carried out to examine the performance of our proposed NS-SPSO algorithm by applying it to 12 widely used benchmark functions. The benchmark functions consist of six uni-modal and six multi-modal problems. The results produced by NS-SPSO are then compared with NS-MJPSO and other state-of-the-art algorithms. The average/best evaluation values are shown in the tables and further illustrated in the graphical figures. The proposed algorithm has consumed shortest computation time and also has produced second best results in terms of accuracy in comparison to all other variants except NS-MJPSO. Furthermore, the proposed algorithm has solved the problem of premature convergence to some extent by the concept of state switching. The particle learns from the population distribution about the neighbourhood and also the global best position. Then the particles efficiently move towards the global optimum in shorter time. The classification of N states has induced the balance between local and global search regions.

The rest of work in this chapter is organized as Section 4.2 is dedicated to summarise the background literature, the structure of basic PSO algorithm and its developments towards the proposed work. In Section 4.3 the structure of the proposed algorithm is presented. In the following Section 4.4 the performance of proposed NS-SPSO algorithm is thoroughly examined in comparison to the other well-known PSO variants. In Section 4.5, we have briefly summarised our proposed work in this chapter.

4.2 Related Work

Particle Swarm Optimization (PSO) is a meta-heuristic, population based algorithm first introduced by Kennedy and Eberhart in 1995 [36; 37]. The main concept is inspired by the swarm intelligent behaviour and choreography of birds flocking and fish schooling [147]. PSO imitates the participant agents called particles to get to the optimum location in the search space. After the random initialization of population the particles compare their current position to their neighbours and thus move to the new position. Basically, each particle is a candidate solution and it keeps track of the best places found by itself within the trial history. It is denoted as personal best or P_{best} symbolically. Whereas, the best value ever found by entire swarm is called global best or G_{best}. All particles are collectively named as swarm. PSO algorithm is based on two simple equations denoted as velocity update v_i, and position update x_i. A constant value 2 is used for the acceleration coefficients c_1 and c_2. Apart from that, two uniformly distributed random numbers denoted as $rand_1$ and $rand_2$ have also been used. The simplified structure, good quality solutions, quick convergence and algorithm reliability are the main characteristics that have attracted researchers in various fields. PSO has been applied to various real world optimization problems [1; 2; 5; 73; 148; 149; 150] in the last two decades. Due to the limitations of getting trapped into local optimum and excessive evaluations the basic PSO has been further modified. Several variants have been developed with extra capabilities.

In PSO modified versions, the diversity of the swarm has been improved by introducing the various structures for topologies in [56; 57]. Kennedy and Mendes in [58] have proposed two different types of topologies named as ring and Von-Neumann topologies. A novel fully-informed PSO (FIPS) has been proposed in [59]. In FIPS the particles learn from their peers with the best fitness in their neighbourhood. Comprehensive learning PSO (CLPSO) has been developed in [4]. This algorithm has also contributed to the area of topological improvements of PSO. The performance of the above mentioned algorithms are investigated on the uni-modal and multi-modal functions. Another variant of PSO is developed that derives the mean distance of particles in the local neighbourhood. This algorithm has been

used for the problems having many local optima [154].

PSO algorithm is used in combination with the other techniques such as evolutionary techniques [7], genetic algorithm [49; 51] and ant bee colony [52]. Additional parameters have been introduced into PSO algorithm. The concept of niching has been incorporated with PSO algorithm in [155]. Gaussian mutation has been introduced by [54]. Another Adaptive Particle Swarm Optimization (APSO) algorithm has been proposed [7], evolutionary state information has been used as an additional term added to the process of basic PSO. Four evolutionary states S_1, S_2, S_3, and S_4 have been introduced. Each state has assigned an appropriate value from the fuzzy membership interval of the particle current state. The evolutionary factor E_f has been used to initialize the population distribution, and to measure the mean distance between the global best and other particles in the swarm. Four states have been described by taking population distribution information E_f in to account, which describes convergence, exploration, exploitation and jumping-out states respectively. Fuzzy classification method is used for classifying the states, which results in some limitations of excessive computation of acceleration coefficients in each generation, swarm stagnation in the local optima, if the current global best is the local optimum and the last one is the complicated implementation of classification method.

Furthermore, in the PSO performance studies, the computation time has been considered as the initial objective for improvement. Another aim that has been considered is solving the problem of local optima or premature convergence [1; 2; 3; 4]. The given improved variants have been thoroughly investigated by applying them into numerous real-world problems. However, due to the non-linear, multi-modal, high-dimensional and complex types of the real-world problems there is still a desirable room for further enhancement to the PSO algorithm. In response to that, the supplementary techniques have been merged to significantly control the parameters of PSO algorithm [6; 7; 8]. The topological structures have been improved to explore the search space, ensure global optimum and avoid premature convergence [4].

A novel hybrid type, switching PSO (SPSO), has been proposed in [8]. Evolutionary state information is used to find the mean distance of all the particles. Then Markov chain is applied to randomly switch particle within the four states according to certain transition probability. Appropriate values of acceleration coefficients have been predefined for all states. The main consideration of the switching PSO is to establish the balance between local, global search region and converge quickly to the global optimum in few iterations. The switching mechanism has ensured that the particle will change its state according to certain probability and will not get trapped into local optimum prematurely. SPSO has shown best performance for benchmark functions and genetic regulatory networks (GRN)

application [8]. Therefore, to make the existing SPSO robust and more accurate, we have proposed some modifications to the current SPSO algorithm.

The proposed N state switching particle swarm optimization algorithm (NS-SPSO) is the modified version of our previously developed algorithm NS-MJSPO described in Chapter 3, where N is number of possible states that can be any positive value. Basically, the main idea is similar to four state versions given in APSO and SPSO [7; 8]. The NS-SPSO algorithm is based on evolutionary techniques. The N states are visualised as sub-states or stages of four states. Furthermore, the evolutionary states are described by calculating and then using the population distribution, mean distance using Euclidean space, maximum minimum values and also the index of global best particle in the population distribution information. The inertia weight ω is an important parameter of PSO algorithm, which has first been introduced by [44]. In this work we compute the inertia weight ω by combing the evolutionary factor and the time varying strategy [46]. Acceleration coefficients c_1 and c_2 both takes N number of tuned values. The proposed algorithm is then applied to 12 commonly used uni-modal and multi-model functions of various dimensions. The results are compared with some well known and most cited algorithms. The proposed algorithm has performed well in terms of shortest computation time and average/best evaluation values in comparison to all variants in comparison except NS-MJPSO in accuracy for most of the benchmark functions. However, in few problems some additional parameter tuning is required to improve the quality of solution in terms of better evaluation values.

4.2.1 The Basic Framework of PSO Algorithm

The PSO algorithm refers to the intelligent searching behaviour of all participants named as particles. The population of all particles is called swarm of size n, where each individual particle i is a candidate solution in the problem space. Each particle i holds two vectors quantities, the first one is the velocity of ith particle in Dth dimension and t time is represented as $v_i(t) = [(v_{i1}(t), v_{i2}(t), \cdots, v_{iD}(t))]$ and the second one is the position of the ith particle in Dth dimension and in time t is denoted as $x_i(t) = (x_{i1}(t), x_{i2}(t), \cdots, x_{iD}(t))$, where D represents dimension of the solution search space. The swarm velocities and positions are initialized randomly with their respective boundaries $x_{in}(t) \in [x_{\min,n}, x_{\max,n}]$ $(1 \leq n \leq D)$ with $x_{\min,n}$ and $x_{\max,n}$ of the search space. Where V_{max} is maximum velocity set to the 20% of the search space [73]. During the process of algorithm evaluation iteratively the particle i with dth dimension is updated as follows.

$$\begin{aligned} v_i(t+1) =& \omega v_i(t) + c_1 rand_1(pbest_i(t) - x_i(t)) \\ &+ c_2 rand_2(gbest_i(t) - x_i(t)) \end{aligned} \tag{4.1}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{4.2}$$

where ω is called inertia weight [44] c_1 and c_2 are denoted as acceleration coefficients [37]. $rand_1$ and $rand_2$ are two uniformly distributed random numbers generated between $U[0,1]$ [36]. *Pbest* represents $pbest_i = (pb_{i1}, pb_{i2}, \cdots, pb_{iD})$. Personal best is particle with the best fitness found by the ith particle so far and *gbest* means *gbest* denoted as $gbest_D = (gb_1, gb_2, \cdots, gb_D)$ global best is the best particle fitness found by the entire swarm. n_{Best} is used for global best in the neighbourhood version. G_{Best} for the global version and L_{Best} for the local version of PSO. The particle's personal experience and its social interaction determines the direction towards its best position iteratively. In [156] the movement of particle in the search space and the influence of its parameter is shown in Figure 4.1.

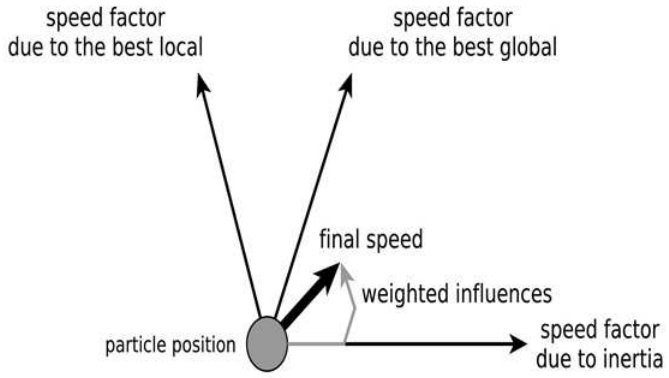


Figure 4.1: The Particle and Parameter Social Learning Behaviour

4.3 The novel N State Switching PSO

This section elaborates the development of novel NS-SPSO for the enhancement of global search performance. A new switching parameter $\delta(t)$ is introduced in the basic PSO velocity update Equation (4.3). The value of N states along-with other parameters is initialised. The novel N state switching PSO (NS-SPSO) is investigated by applying to 12 uni-modal and multi-modal widely used benchmark functions [4; 153] which are given in Section 4.4 and Table 3.1.

$$\begin{aligned} v_i(t+1) =& \omega v_i(t) + c_1(\delta(t))r_1(t)(pbest_i(t) - x_i(t)) \\ &+ c_2(\delta(t))r_2(t)(gbest_i(t) - x_i(t)), \end{aligned} \tag{4.3}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{4.4}$$

4.3.1 Prediction of Evolutionary States

In the beginning of population distribution the particles are dispersed in the search space. However, in the evolutionary process the particles group together iteratively in the later stages and find their local and global optimal positions in the search space. The extraction of information from the population distribution and using that for further describing the evolutionary state is an important research topic in PSO. Hence, the population distribution information in each generation is important to be recorded. A clustering based technique was introduced for evolutionary state estimation in [6; 143]. Whereas, fuzzy classification method is used for calculating four evolutionary states in [7].

In the first step of population distribution the mean distance from the global best particle in the search space for each i is derived. The particles having smaller distance from the global best are close to the convergence state and it switch to the other state according to evolutionary factor. Furthermore, the particles located far away from the global best switch to another state with higher values of its parameters. The mean distance is calculated by using Euclidean matrix as follows [7; 8]:

$$P_d(i) = \frac{1}{N-1} \sum_{j=1, j\neq i}^{N} \sqrt{\sum_{k=1}^{D} (x_i(k) - \bar{x}_j(k))^2} \tag{4.5}$$

In Equation (4.5) N represents swarm size and D stands for dimensions of the problem. Evolutionary factor E_f has been introduced by [7], and it has further been used by [8].

This thesis presents a novel switching mechanism by extending from four states up to N states. It further divides the four states to possible sub-states. The sub-states smoothly describe the unit of association for particle in a particular state. By dividing into sub-states, we assume significant improvement in adopts more suitable values for its parameters. The sub-states represents the certain stages according to the value of N states. Hence, by increasing the number of states the performance of the algorithm will be improved in terms of accuracy in the evaluation results. But the computation burden will increase slightly. An auxiliary parameter δ is used in the new velocity update Equation (4.3) in Section 4.3.

Here we have derived the mean distance of all $P_d(i)$ by using Equation (4.5) and find P_{dg} the

global best particle, $P_{d(max)}$ as the maximum mean distance and $P_{d(min)}$ as the minimum mean distances. Consider the values derived by using Equation (4.5) in the population distribution and then compute the evolutionary factor using Equation (4.6).

$$E_f = \frac{P_{dg} - P_{d(min)}}{P_{d(max)} - P_{d(min)}} \in [0, 1] \tag{4.6}$$

$$States = \begin{cases} 1, & 0 \leq E_f < \frac{1}{N}, \\ 2, & \frac{1}{N} \leq E_f < \frac{2}{N}, \\ 3, & \frac{2}{N} \leq E_f < \frac{3}{N}, \\ \vdots & \vdots \\ N, & \frac{N-1}{N} \leq E_f < 1 \end{cases}$$

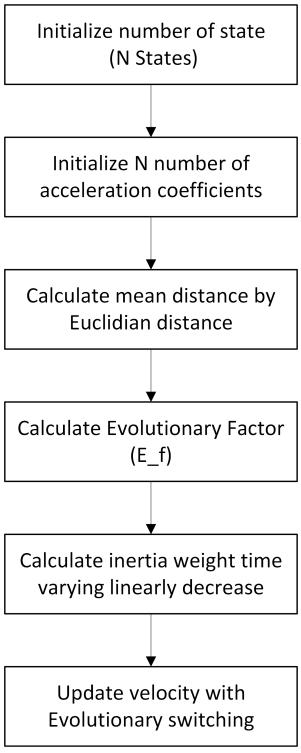


Figure 4.2: The switching parameters flow diagram

4.3.2 Mechanism for Inertia Weight Calculation

The inertia weight ω has the significant contribution in the control of PSO algorithm. It has the main influence on the global and local search performance. If the value of ω is small then it causes exploitation in the local search region. Large ω drag the swarm towards global search region. In this chapter ω is by using the linearly decreasing strategy [46].

$$\omega = (\omega_{max} - \omega_{min}) \times \frac{\text{iter}}{\text{maxiter}} + \omega_{min} \tag{4.7}$$

Here, we have initialized $\omega = 0.9$ as maximum and 0.5 values as minimum. The value of inertia weight ω is linearly decremented with time. The developments regarding inertia weight has been briefly described in Chapter 3. The complete structure of NS-SPSO is described by the following flowchart in Figure 4.3:

4.3.3 Selection of Acceleration Coefficients

In the proposed NS-SPSO the acceleration coefficients are selected and adjusted manually according to the problem. N number of acceleration coefficients are required. For instance, if $N = 4$ then we have to initialize four values for each acceleration coefficient $C = [2, N]$. Each value is designated to a particular state. The N acceleration coefficient values are pre-initialized. Initially, $c_1(\delta(0))$ and $c_2(\delta(0))$ are assigned 2. Then the appropriate value for the acceleration coefficient are automatically assigned during the program execution time. The strategy for selecting the acceleration coefficients for each state is described in [8] as follows:

In the proposed NS-SPSO the large value of E_f describes the state as *jumping-out-state*. As the particle has the intention to jump from the local optimum towards global optimum a large value of social learning factor $c_2(\delta(4)) = 2.2$ and smaller value of cognitive factor $c_1(\delta(4)) = 1.8$ are assigned. Subsequently, the particle flies towards the global best region very quickly. According to this strategy the proposed algorithm converges to global optimum in few iterations. Similarly, a relatively small value of E_f describes the current state in the *exploration-state*, according to that a large value of $c_1(\delta(3)) = 2.2$ and smaller value of $c_2(\delta(3)) = 1.8$ are assigned to let the particle explore search spaces on its personal influence. Moreover, in the *exploitation-state*, a large value of $c_1(\delta(2)) = 2.1$ and smaller value of $c_2(\delta(2)) = 1.9$ are pre-initialized. A slight changes have been made to preserve the balance in local and global search performance. Subsequently, in the *convergence-state*

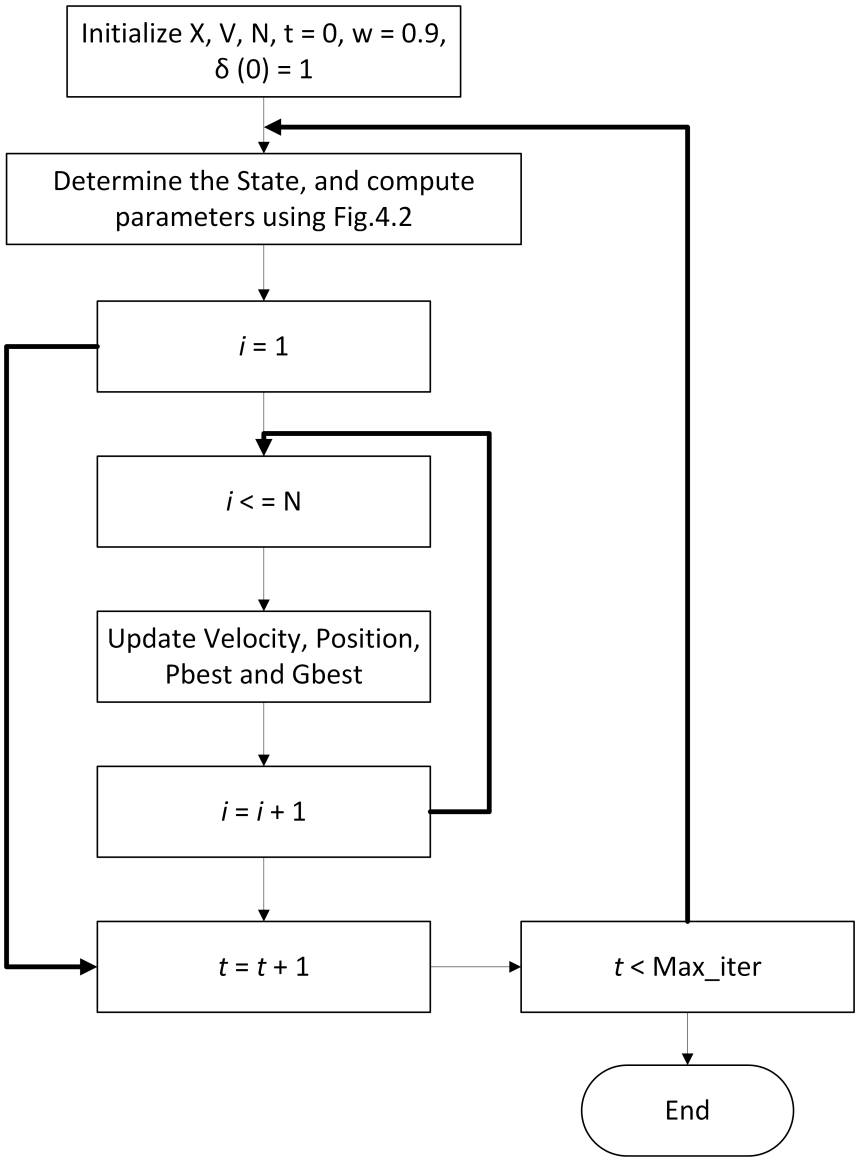


Figure 4.3: N State Switching PSO Algorithm flowchart

equal values to both $c_1(\delta(1)) = 2.0$ and $c_2(\delta(1)) = 2.0$ because all particles group together in the convergence-state.

4.4 The Experimental Work

The proposed NS-SPSO has been evaluated over 12 commonly used benchmark functions that are given in Table 3.1 f_1 to f_{12} taken from [14]. Initially, the proposed NS-SPSO is applied to the 12 functions f_1 to f_{12} in 30 dimensions. Then the evaluation results are compared with published values of six state-of-the-art algorithms. All the variants are re-implemented and evaluated for 30 independent trials. The published results of all the variants are taken from [14] for comparison. All the experimental work have been conducted on a PC with an Intel Core i5-3320M 2.6 GHz CPU and Microsoft Windows 10 Pro 64-bit system. The benchmark test experiments of the 12 uni-modal and multi-modal problems for the proposed NS-SPSO and other PSO variants in comparison are all coded in Matlab R2015a. It is also worth to be mentioned that because of the evolutionary control and switching techniques the algorithm converge to its optimum in the early stages of the function evaluations.

Table 4.1: Parameter coefficients of the PSO variants for comparison

Algorithm	Inertia weight	Acceleration Coefficients
LPSO	$[0.9, 0.4]$	$c_1 = c_2 = 2.0$
GPSO	$[0.9, 0.4]$	$c_1 = c_2 = 2.0$
DMS-PSO	0.729	$c_1 = c_2 = 1.49445, m = 3, R = 15$
FIPS	$\chi = 0.729$	$c_1 + c_2 = 4.1$
CLPSO	$[0.9, 0.7]$	$c_1 = c_2 = 1.49445$
NS-MJPSO	$[0.9, 0.5]$	$c_1 = [2, N], c_2 = [2, N], \phi = 0.9, N$
NS-SPSO	$[0.9, 0.5]$	$c_1 = [2, N], c_2 = [2, N], N$

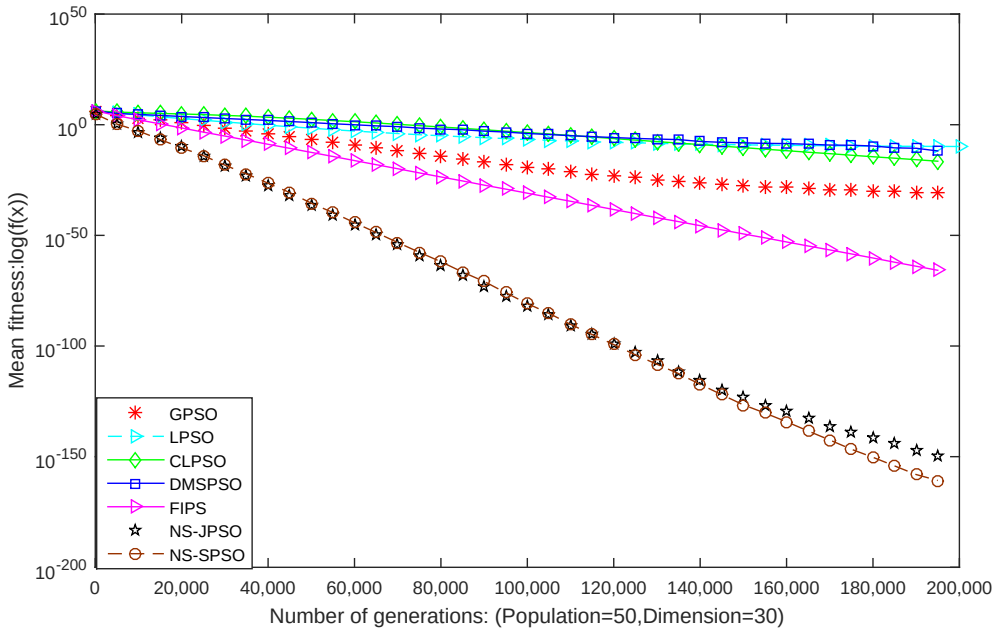
4.4.1 Performance Analysis of NS-SPSO in Benchmark Functions

To analyse the performance of proposed NS-SPSO algorithm in benchmark functions, we compare it to the newly developed NS-MJPSO Chapter 3 and five other variants of PSO algorithm. All the variants have been re-implemented for comparison purposes. The published values are used in the tables here produced by [14]. We compare our newly developed NS-MJPSO algorithm described in Chapter 3 because we aim to further enhance its performance by reducing the computational burden. Second variant is the local-neighbourhood

PSO (Local-PSO) [58], third is the global best version (GPSO) [46], fourth is the dynamic multi-swarm version of PSO (DMS-PSO) [60; 62], fifth is the fully-informed PSO (FIPS) [59], the sixth and last one is the comprehensive-learning PSO (CLPSO) [4]. The required parameters along-with the values are described here in Table 4.1. The proposed NS-SPSO has characterised outstanding performance on 9 out 12 problems (f_1 to f_6 and f_8 to f_{12}) containing uni-modal and multi-modal problems. We have shown the performance of proposed NS-SPSO algorithm individually for each function in Table 4.2 to Table 4.13 and Figure 4.4 to Figure 4.15 as follows:

Table 4.2: (f_1) The Sphere function

Algorithm	Mean	Best Value	Std Dev
NS-SPSO	6.86E-42	1.71E-161	2.28E-41
NS-MJPSO	2.16E-150	2.79E-161	1.82E-144
LPSO	4.89E-12	1.80E-14	4.86E-12
FIPS	7.23E-70	4.78E-71	6.55E-70
DMS-PSO	3.81E-15	4.97E-20	1.02E-14
CLPSO	6.32E-19	1.69E-19	4.56E-19
GPSO	5.56E-33	3.30E-45	1.93E-32



(f_1) The Sphere Function

Figure 4.4: The mean fitness of f_1 30 dimensional problems

We have executed the algorithms for 30 independent trails due to the randomness of the

algorithms results. In the empirical results we store the mean, small evaluation errors and standard deviation for each function in all trails. In Table 4.2 to Table 4.13 the statistical results are given over 30 independent trails for the newly developed NS-SPSO and all other comparison algorithms.

Sphere function (f_1) is a uni-modal and simplest function surrounded by all other functions. In Table 4.2 and Figure 4.4 demonstrates the performance of all the algorithms. The newly developed NS-SPSO has produced the best value in comparison to all other algorithms in terms of the minimum values. The NS-SPSO is also faster in execution in comparison. Therefore, the newly developed NS-SPSO has the promising performance for uni-modal problems. NS-SPSO is on top rank for (f_1). Furthermore, NS-MJPSO is on rank 2 for (f_1).

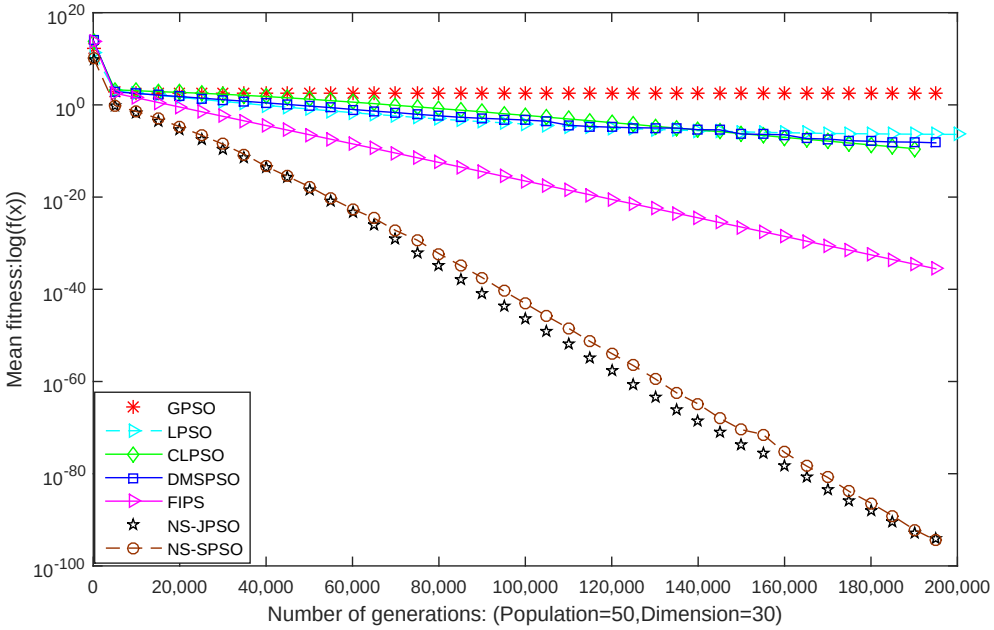
Table 4.3: (f_2) The Schwefel 2.22 Function

Algorithm	Mean	Best Value	Std Dev
NS-SPSO	4.67E+00	3.46E-95	6.29E+00
NS-MJPSO	7.93E-95	2.51E-98	6.40E+00
LPSO	1.33E-08	9.36E-10	1.39E-08
FIPS	9.99E-39	2.71E-39	5.40E-39
DMS-PSO	3.29E-11	1.42E-14	8.70E-11
CLPSO	7.49E-12	4.70E-12	2.28E-12
GPSO	9.67E+00	1.85E-28	1.03E+01

Schwefels function 2.22 (f_2), is also uni-modal and simple function amongst all other functions. In Table 4.3 and Figure 4.5 the performance of all the algorithms have illustrated. The newly developed NS-MJPSO has produced the best value in comparison to all other algorithms in terms of the minimum values. The NS-SPSO is also faster in execution in comparison. However, the newly developed NS-SPSO and NS-MJPSO algorithms have the promising performance for uni-modal problems. NS-MJPSO is on top rank for (f_2). NS-SPSO is on rank 2 for (f_2).

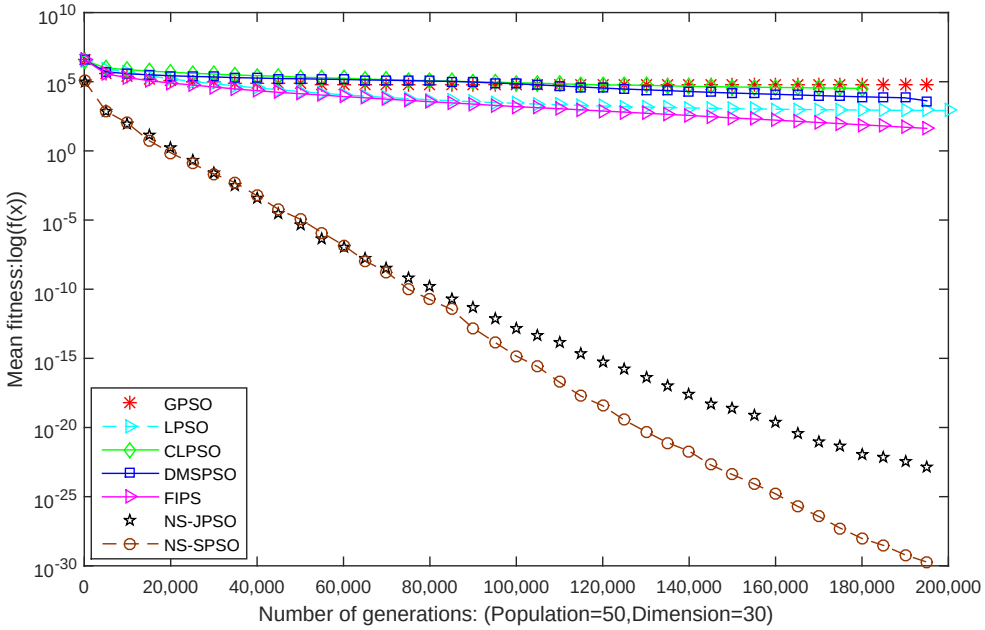
Table 4.4: (f_3) The Schwefel 1.2 Function

Algorithm	Mean	Best Value	Std Dev
NS-SPSO	1.67E+02	1.87E-30	9.13E+02
NS-MJPSO	1.48E-23	5.45E-27	9.13E+02
LPSO	2.75E+01	8.10E+00	1.43E+01
FIPS	1.16E+00	3.58E-01	6.05E-01
DMS-PSO	8.35E+01	1.06E+01	5.51E+01
CLPSO	1.06E+03	6.74E+02	3.20E+02
GPSO	2.22E+03	4.44E-05	3.46E+03



(f_2) The Schwefel 2.22 Function

Figure 4.5: The mean fitness of f_2 on 30 dimensional problems



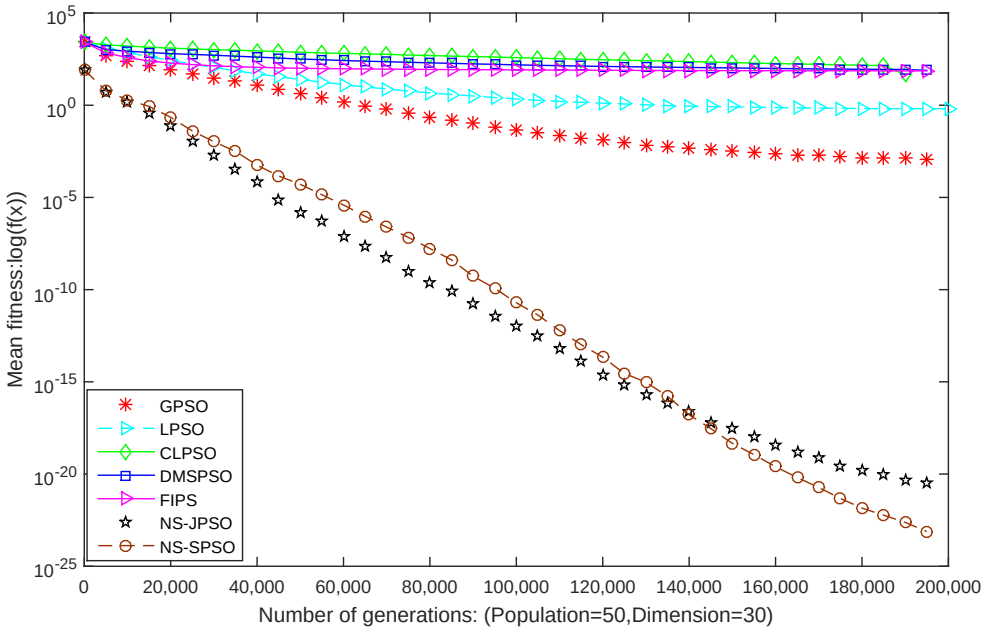
(f_3) The Schwefel 1.2 Function

Figure 4.6: The mean fitness of f_3 30 dimensional problems

Schwefels function 1.2 (f_3), is a unimodal and simple problem amongst other functions. In Table 4.4 and Figure 4.6, we have presented the performance of all the algorithms. The newly developed NS-SPSO has produced the best value in comparison to all other algorithms in terms of the minimum values. The NS-SPSO is also faster in execution in comparison. However, the newly developed NS-SPSO and NS-MJPSO algorithms have the promising performance for uni-modal problems. NS-SPSO is on top rank for (f_3). NS-MJPSO is on rank 2 for (f_3).

Table 4.5: (f_4) The Schwefel 2.21 Function

Algorithm	Mean	Best Value	Std Dev
NS-SPSO	1.65E-05	7.63E-24	2.78E-05
NS-MJPSO	3.04E-21	1.48E-23	8.22E-21
LPSO	2.14E-02	8.23E-03	8.04E-03
FIPS	2.42E+00	3.60E-01	1.15E+00
DMS-PSO	2.14E+00	8.52E-01	8.80E-01
CLPSO	4.50E+00	3.32E+00	5.15E-01
GPSO	3.87E-05	3.71E-06	3.27E-05



(f_4) The Schwefel 2.1 Function

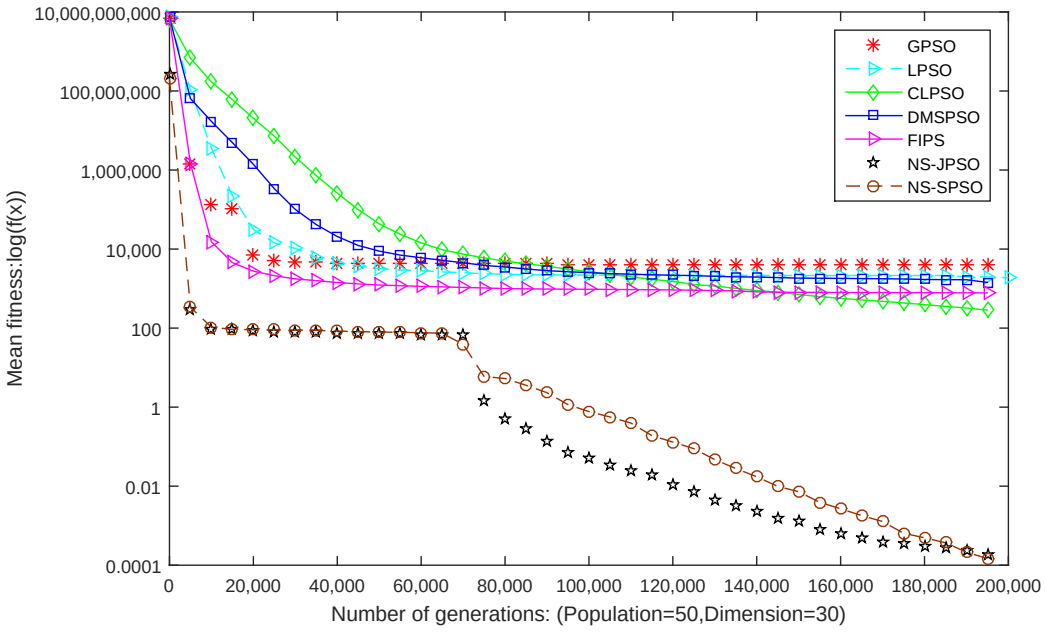
Figure 4.7: The mean fitness of f_4 on 30 dimensional problems

Schwefels function 2.21 (f_4) is a uni-modal and simple problem amongst other functions. Table 4.5 and Figure 4.7 describes the best results in terms of accuracy and convergence

speed for all algorithms. The newly developed NS-SPSO has produced the best value in comparison to all other algorithms in terms of the minimum values. The NS-SPSO is also faster in execution in comparison. However, the newly developed NS-SPSO and NS-MJPSO algorithms have the promising performance for uni-modal problems. NS-SPSO is on top rank for (f_4). NS-MJPSO is on rank 2 for (f_4).

Table 4.6: (f_5) The Rosenbrock Function

Algorithm	Mean	Best Value	Std Dev
NS-SPSO	1.14E+01	1.47E-04	1.75E+01
NS-MJPSO	1.92E-04	8.47E-09	1.64E+04
LPSO	6.27E+01	7.32E+00	5.98E+01
FIPS	2.59E+01	1.25E-01	1.71E+01
DMS-PSO	3.86E+01	2.74E-01	3.03E+01
CLPSO	9.55E+00	1.73E+00	7.73E+00
GPSO	1.31E+02	3.84E-01	5.59E+02



(f_5) The Rosenbrock Function

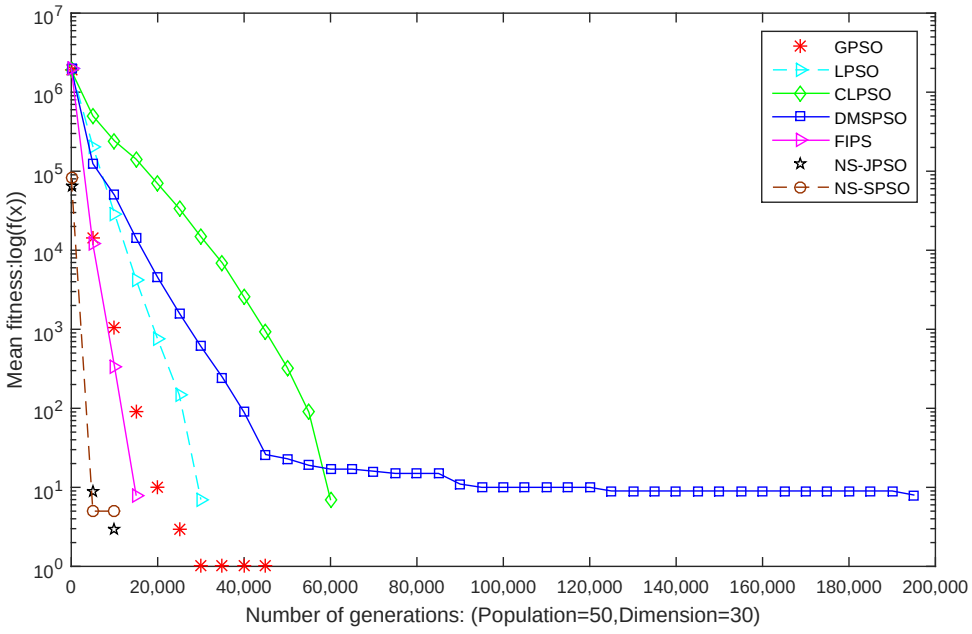
Figure 4.8: The mean fitness of f_5 30 dimensional problems

Rosenbrock function (f_5), is a unimodal problem. In Table 4.6 and Figure 4.8 the performance of all the algorithms have been given. The newly developed NS-MJPSO has produced the best value in comparison to all other algorithms in terms of the minimum

values. The NS-SPSO is the fastest one in execution time in comparison to all algorithms. However, the newly developed NS-SPSO and NS-MJPSO algorithms have the promising performance for uni-modal problems. NS-MJPSO is on top rank for (f_5). NS-SPSO is on rank 2 for (f_5).

Table 4.7: (f_6) The Step Function

Algorithm	Mean	Best Value	Std Dev
NS-SPSO	0.00E+00	0.00E+00	0.00E+00
NS-MJPSO	0.00E+00	0.00E+00	0.00E+00
LPSO	0.00E+00	0.00E+00	0.00E+00
FIPS	0.00E+00	0.00E+00	0.00E+00
DMS-PSO	2.67E-01	0.00E+00	5.21E-01
CLPSO	0.00E+00	0.00E+00	0.00E+00
GPSO	0.00E+00	0.00E+00	0.00E+00



(f_6) The Step Function

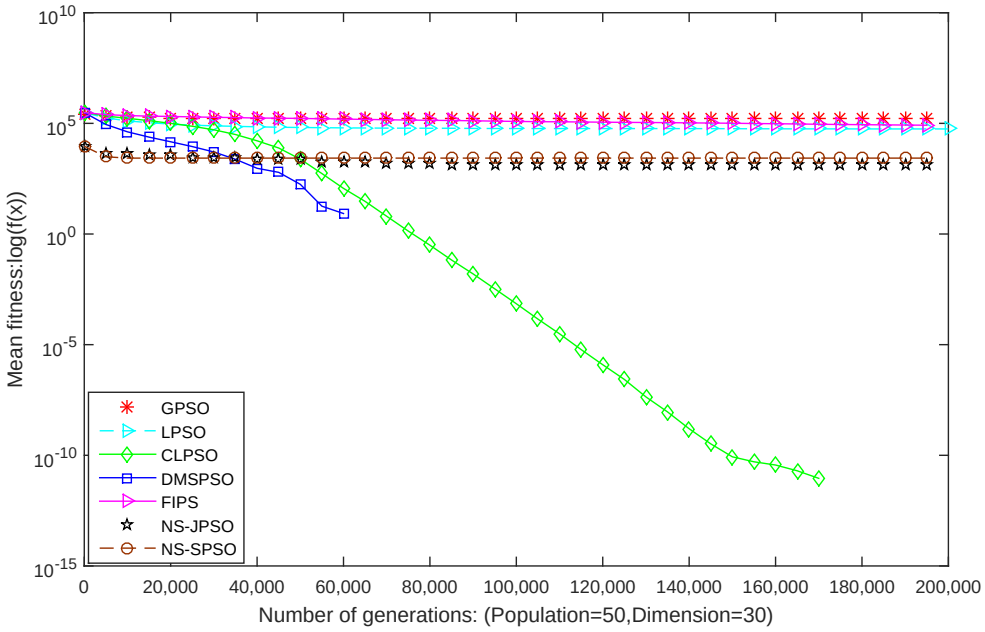
Figure 4.9: The mean fitness of f_6 on 30 dimensional problems

Step function (f_6), is a discontinuous function, it is also called piecewise constant function. In Table 4.7 and Figure 4.9, we have plotted the simulation results, which further describes the best performance of all the algorithms. All the algorithms have produced the same global optimum. However the newly developed NS-SPSO is the fastest one in execution

time in comparison to all algorithms. However, the newly developed NS-SPSO and NS-MJPSO algorithms have the promising performance for such kind of problems.

Table 4.8: (f_7) The Schwefel Function

Algorithm	Mean	Best Value	Std Dev
NS-SPSO	6.16E+03	2.69E+03	2.04E+03
NS-MJPSO	2.99E+03	1.40E+03	2.16E+03
LPSO	1.87E+03	9.51E+02	5.30E+02
FIPS	2.70E+03	1.34E+03	7.56E+02
DMS-PSO	-2.55E-01	-7.66E+00	1.40E+00
CLPSO	4.85E-13	0.00E+00	8.18E-13
GPSO	5.52E+03	2.82E+03	2.33E+03



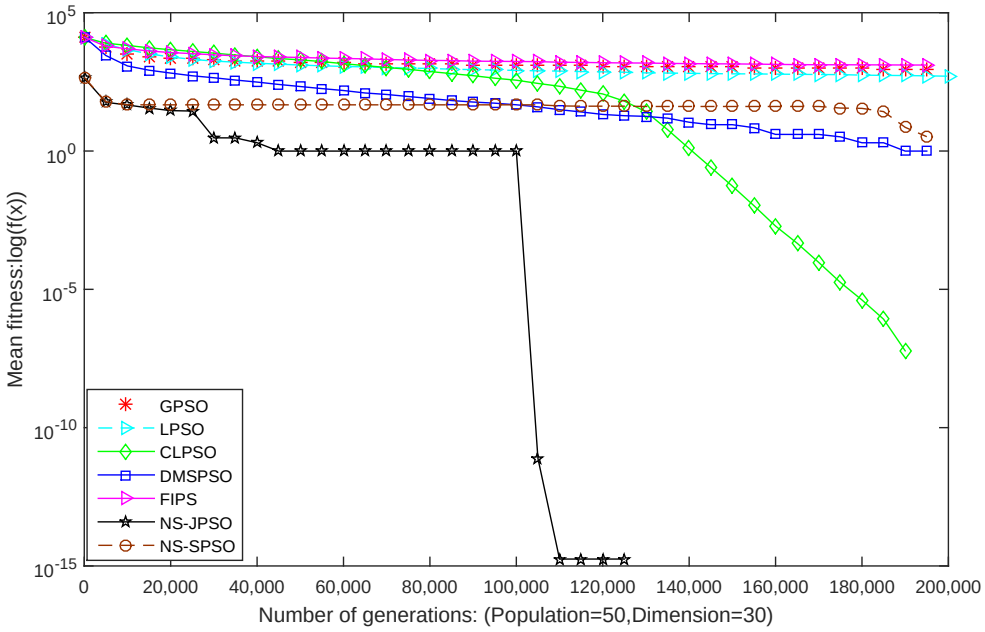
(f_7) The Schwefel Function

Figure 4.10: The mean fitness of f_7 30 dimensional problems

Schwefels function (f_7), is a multimodal problem. In Table 4.8 and Figure 4.10 the numerical and graphical results have been illustrated for all the algorithms. CLPSO and DMS-PSO have the minimum values for this multimodal function. The newly developed NS-SPSO and NS-MJPSO needs some further parameters adjustment to produce good results. However, in the same setting the performance is not good for this function.
Rastrigin function (f_8), is a highly multi-modal problem. In Table 4.9 and Figure 4.11

Table 4.9: (f_8) The Rastrigin Function

Algorithm	Mean	Best Value	Std Dev
NS-SPSO	2.48E+01	3.30E+00	2.19E+01
NS-MJPSO	0.00E+00	0.00E+00	0.00E+00
LPSO	1.69E+01	4.01E+00	1.05E+01
FIPS	4.25E+01	2.70E+01	6.66E+00
DMS-PSO	3.32E-02	1.78E-15	1.82E-01
CLPSO	6.13E-09	4.23E-10	6.86E-09
GPSO	3.05E+01	1.98E-07	3.15E+01



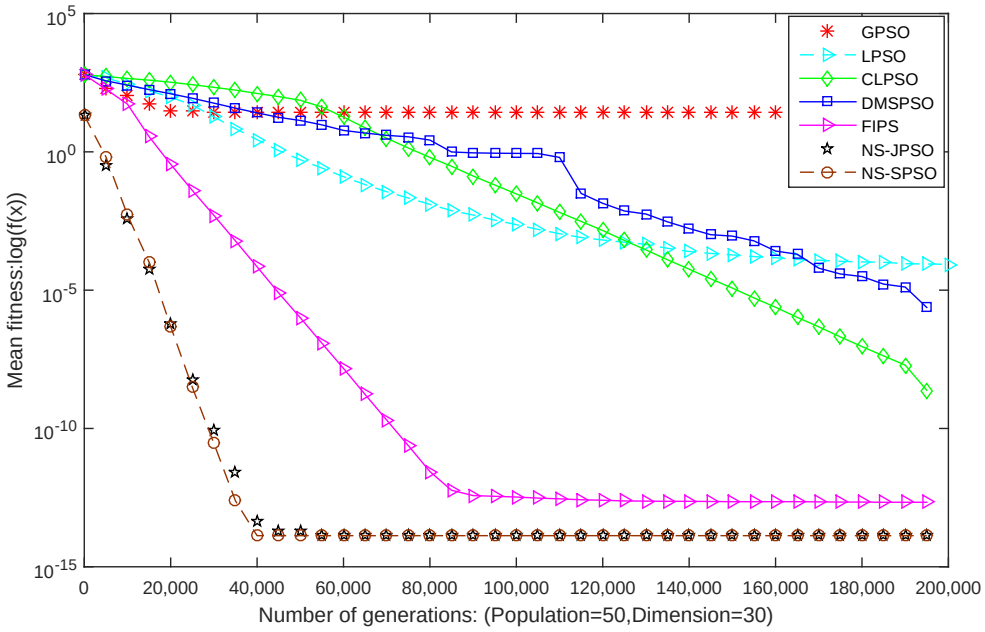
(f_8) The Rastrigin Function

Figure 4.11: The mean fitness of f_8 on 30 dimensional problems

shows the average/best performance of all the algorithms. NS-MJPSO, CLPSO, DMS-PSO and GPSO have the best performance for this multi-modal function. The newly developed NS-SPSO needs some further parameters adjustment to get good results.

Table 4.10: (f_9) The Ackley's Function

Algorithm	Mean	Best Value	Std Dev
NS-SPSO	1.45E-14	1.33E-14	3.28E-15
NS-MJPSO	1.33E-14	6.22E-15	3.41E-15
LPSO	2.87E-06	7.68E-08	5.48E-06
FIPS	7.16E-15	6.22E-15	2.46E-15
DMS-PSO	1.49E-08	6.84E-11	3.71E-08
CLPSO	2.98E-10	1.24E-10	9.33E-11
GPSO	9.25E-01	6.22E-15	3.52E+00



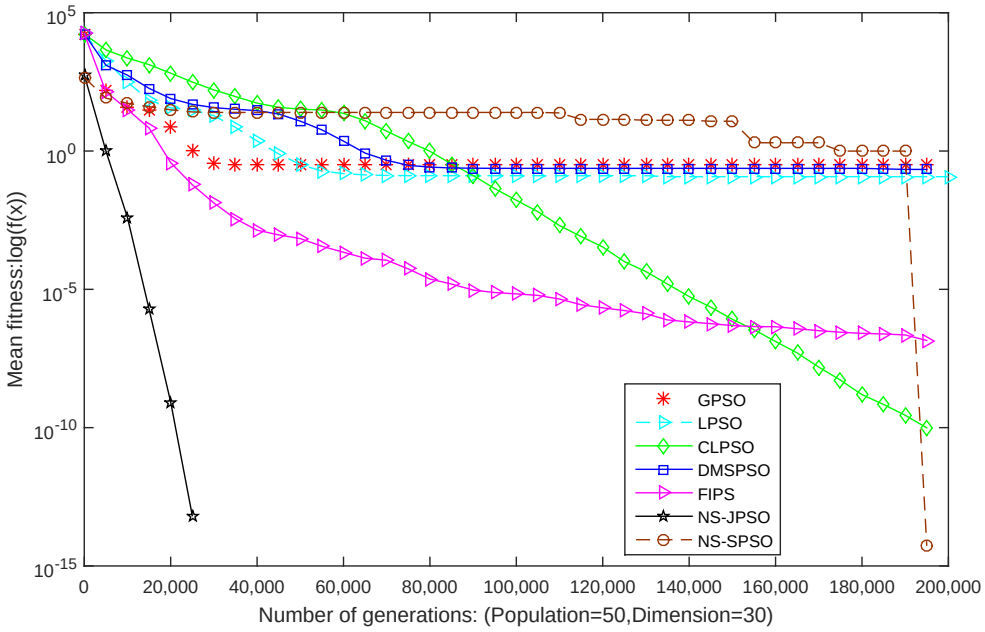
(f_9) The Ackley's Function

Figure 4.12: The mean fitness of f_9 30 dimensional problems

Ackleys function (f_9), is a multi-modal and separable function. In Table 4.10 and Figure 4.12, we have plotted the numerical and graphical illustrations for all algorithms. NS-SPSO, NS-MJPSO, CLPSO, DMS-PSO and GPSO have the similar best performance in terms of best evaluation values. NS-SPSO and NS-MJPSO are relatively faster than the other algorithms.

Table 4.11: (f_{10}) The Griewank Function

Algorithm	Mean	Best Value	Std Dev
NS-SPSO	9.74E-02	5.33E-15	4.02E-01
NS-MJPSO	0.00E+00	0.00E+00	0.00E+00
LPSO	3.94E-03	2.44E-13	6.29E-03
FIPS	4.48E-09	0.00E+00	2.34E-08
DMS-PSO	7.22E-03	0.00E+00	1.06E-02
CLPSO	2.20E-12	7.22E-15	4.28E-12
GPSO	1.05E-02	0.00E+00	1.44E-02



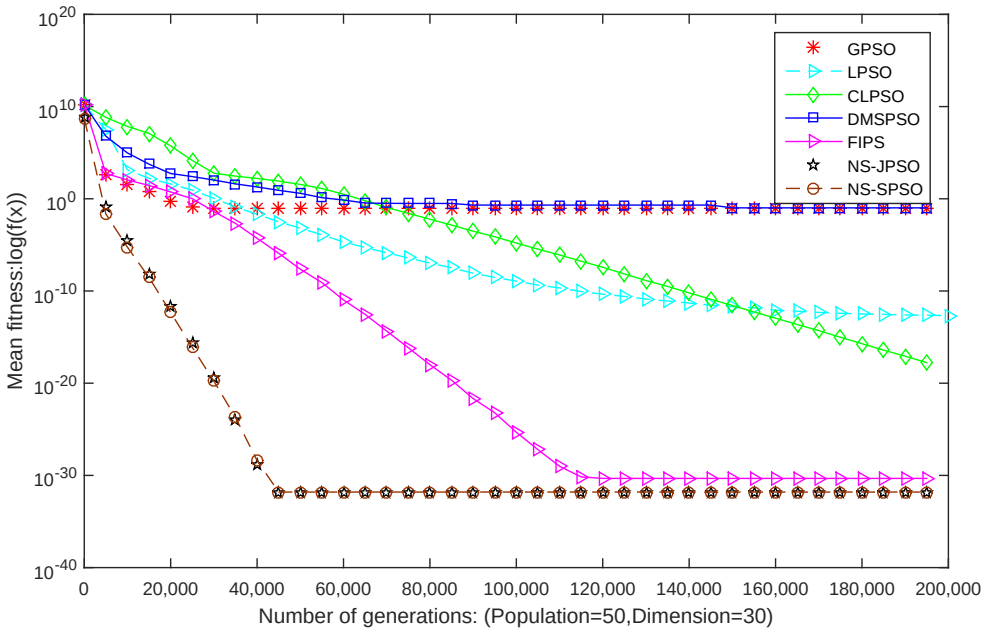
(f_{10}) The Griewank Function

Figure 4.13: The mean fitness of f_{10} on 30 dimensional problems

The Griewank function (f_{10}), is a multi-modal function. Table 4.11 and Figure 4.13 have further described the average/best performance of all the algorithms. NS-SPSO, NS-MJPSO, GPSO and CLPSO have the similar best performance in terms of best evaluation values. NS-SPSO and NS-MJPSO are relatively faster than the other algorithms.

Table 4.12: (f_{11}) The Penalized 1 Function

Algorithm	Mean	Best Value	Std Dev
NS-SPSO	4.15E-02	1.57E-32	8.86E-02
NS-MJPSO	1.57E-32	1.57E-32	5.76E-02
LPSO	6.97E-15	2.45E-16	1.20E-14
FIPS	1.57E-32	1.57E-32	5.57E-48
DMS-PSO	3.46E-03	1.24E-21	1.89E-02
CLPSO	3.09E-20	9.08E-21	1.56E-20
GPSO	3.46E-03	1.57E-32	1.89E-02



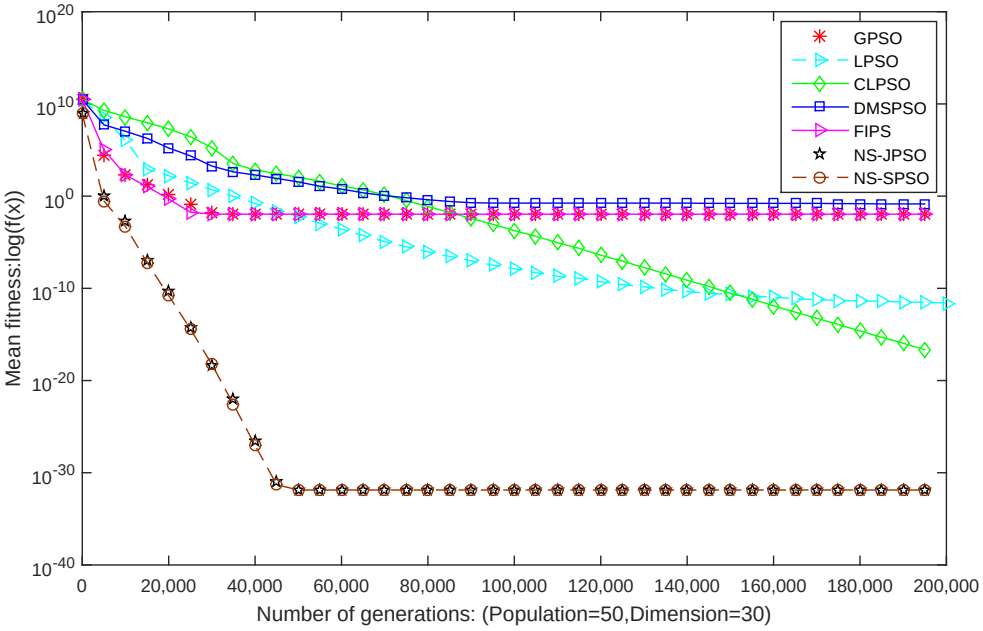
(f_{11}) The Penalized 1 Function

Figure 4.14: The mean fitness of f_{11} 30 dimensional problems

The Penalized 1 function (f_{11}), is a multi-modal function. In Table 4.12 and Figure 4.14 the results of all the algorithms have been described. NS-SPSO, NS-MJPSO, GPSO and FIPS have the same best performance in terms of best evaluation values. NS-SPSO and NS-MJPSO are relatively faster than the other algorithms.

Table 4.13: (f_{12})The Penalized 2 Function

Algorithm	Mean	Best Value	Std Dev
NS-SPSO	2.20E-03	1.35E-32	4.47E-03
NS-MJPSO	1.35E-32	1.35E-32	3.35E-03
LPSO	8.45E-14	3.36E-15	1.27E-13
FIPS	3.66E-04	1.35E-32	2.01E-03
DMS-PSO	4.76E-03	1.47E-18	8.98E-03
CLPSO	4.17E-19	1.09E-19	2.82E-19
GPSO	1.43E-16	1.35E-32	7.82E-16



(f_{12}) The Penalized 2 Function

Figure 4.15: The mean fitness of f_{11} 30 dimensional problems

The Penalized 2 function (f_{12}), is a multi-modal function. In Table 4.13 and Figure 4.15 the evaluation errors and convergence have been demonstrated of all the algorithms. NS-SPSO, NS-MJPSO, GPSO and FIPS have the same best performance in terms of best evaluation values. NS-SPSO and NS-MJPSO are relatively faster than the other algorithms.

4.4.2 Computation Time of the Proposed NS-SPSO

In this section we demonstrate the average computation for our proposed NS-SPSO algorithm. All the algorithm are executed with same number of iterations i.e. 2×10^5 for the same 12 functions. The mean time for all the algorithms in shown in Figure 4.16. The Local PSO (LPSO) has the smaller average computation time. The proposed NS-SPSO algorithm has the second shortest computation time and also having the second average/best minimum values for most of the uni-modal and multi-modal problems in terms of accuracy.

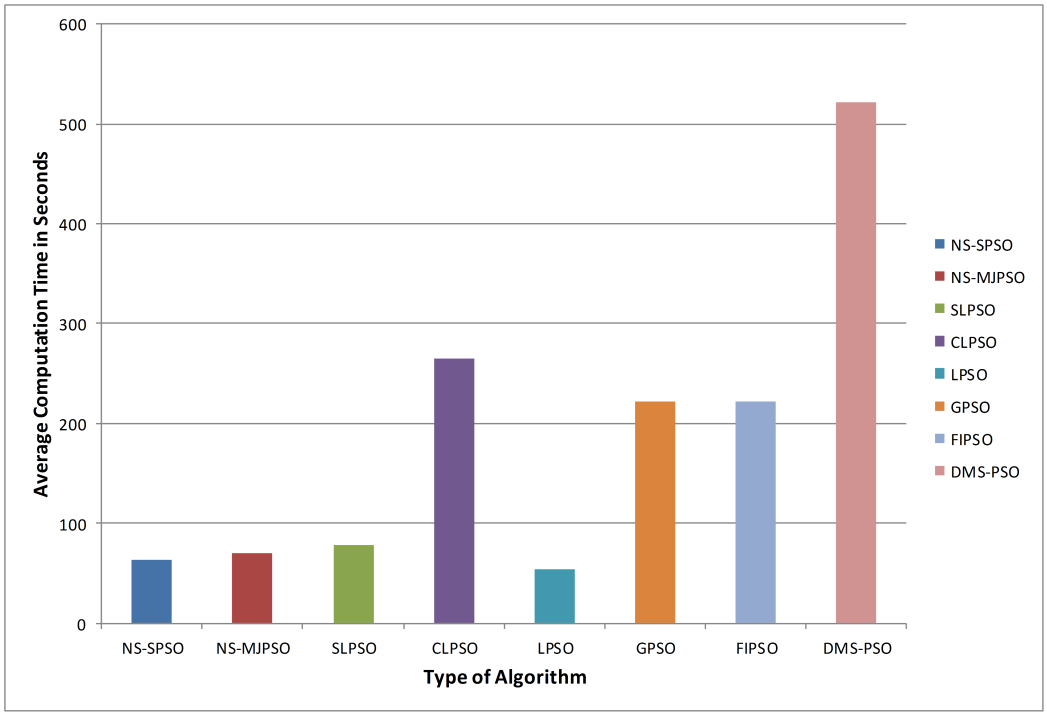


Figure 4.16: Average computation of the proposed NS-SPSO in comparison

4.5 Summary

In this chapter we have proposed another novel N State Switching Particle Swarm Optimization (NS-SPSO) algorithm. That has combined the evolutionary method for popula-

tion distribution with particle swarm optimization (PSO) algorithm. The performance of newly developed NS-SPSO algorithm is described by evaluating 12 benchmark functions. The benchmark functions include some uni-modal, multi-modal and non-linear type problems. Some variants of PSO are re-implemented, which are the most common for their best performance and capabilities. The variants are Globalbest PSO (GPSO), Localbest PSO(LPSO), Fully-informed PSO (FIPS), Dynamic Multi-Swarm PSO (DMS-PSO) and Comprehensive-learning PSO (CL-PSO). The newly developed NS-MJPSO is also used in the tournament for the same objective functions. The significance of proposed work is described by analysing the statistical results in tables and figures. The proposed NS-SPSO has produced shortest computation time and second average/best results in terms of accuracy.

Chapter 5

Economic Load Dispatch Using Novel Particle Swarm Optimization Algorithms

5.1 Introduction

This chapter has contributed a successful application of two newly developed N State Markov Jumping Particle Swarm Optimization (NS-MJPSO) and N State Switching Particle Swarm Optimization (NS-SPSO) in the power system operations. In the power system operations we have selected Economic Load Dispatch (ELD) as the objective problem for our consideration. The novel NS-MJPSO algorithm is investigated by applying it to different type of smooth and non-smooth cost functions of ELD problems. The ELD with smooth function is based on the simplified quadratic function. While the non-smooth function have some constraints to be satisfied. The ELD with smooth cost function is easy to solve by mathematical methods. However, the mathematical models are find difficult to solve the non-smooth functions with non-linear constraints. In this chapter, we have considered ELD with simplified objective and also ELD with valve-point loading effects. The remaining work is outlined as follows: In Section 5.2, we summarise some related work about PSO applications in ELD problems. Section 5.3, we present problem formulation for the ELD problems. In Section 5.4, the implementation setting and procedure is described. The newly developed algorithms in thesis, named, NS-MJPSO, and NS-SPSO, alog-with other well-known variants of PSO are implemented. In Section 5.5, extensive simulations are carried out and the comparative results are plotted. In Section 5.6 the complete work in this chapter is summarised.

5.2 Related Work

In power system operations, Economic Load Dispatch (ELD) is considered as the ultimate topic of optimization. Basically, in essence, the objective of ELD is to decrease the overall generation cost of power systems, whereas the constraints within the systems are satisfied. Prior to the current methods the conventional predictable methods have been applied to solve ELD problems. Due to the exponential increase in power demand has diverted the interest of researchers to develop some modern optimization methods for power system operation ELD problems to attain the maximum production of the existing power systems units. According to the non-linear nature of ELD objective such valve-point loading effects, non-smooth or convex type of system constraints such as prohibited operating zones, ramp-rate limits and high-dimensionality. The conventional method are found to be most of the time infeasible to solve the problem [10; 121]. Numerous efforts have been done to dispatch power to the loads economically and reliably.

Subsequently, Artificial Intelligence (AI) techniques have been applied to solve ELD

problems. In [157] Hopfield method is applied to ELD problems with prohibited operating zones constraints and smooth cost functions. However, the Hopfield method failed to work because of the large number of computations. Genetic Algorithm (GA) has been applied, which is a population based evolutionary technique for global optimization [158]. Another probabilistic and heuristic technique named Simulated Annealing (SA) has been applied in power system operations [159]. The GA algorithm has been reported as the fastest technique due to parallel concept of searching. It has widely been used in solving power system especially for ELD problems. The feasibility of GA has been analysed during solving the application of ELD considering all the non-linear type of constraints. However, the main disadvantage is the requirement of excessive iterations [160; 161].

Particle Swarm Optimization (PSO) algorithm has been applied in power system for reactive power with security constraints in [109]. State estimation is also one of the key power system practical problems. A variant of PSO algorithm is developed in combination with the basic selection mechanism of Genetic Algorithm (GA). The resultant variant is called Hybrid PSO (HPSO) [162]. Another modified version is PSO has been applied for parameters optimization in power systems stabilizers [40].

PSO has been proposed and applied for practical ELD problems with non-linear constraints [10]. The non-linear type constraints includes prohibited operating zones, ramp-rate limits and non-smooth objective refer to the cost function with valve-point loading effects. It has been revealed in the results that PSO algorithm performed better than GA in comparison. In [163] PSO has been employed to solve ELD problem considering the voltage and line flow constraints with objective. The distinctive best performance of proposed PSO algorithm has been examined by comparing its results with conventional methods and GA applied to same problem.

In [164] a new PSO (NPSO) algorithm has been developed with the split-up mechanism for personal influence or cognitive factor. As in the traditional PSO algorithm the particle keeps track of the best location found in the history. The proposed NPSO keeps track of the worst position as well so that to better explore the search space. Furthermore, a Local Random Search (LRS) technique is also combined with NPSO. Later on, the NPSO-LRS has been applied for non-smooth ELD problems in power systems. The best performance has been observed and presented.

PSO algorithm has been modified by incorporating it with the Gaussian Probability Distribution [122]. The subsequent PSO algorithm is then applied to ELD problem in power systems. All generator constraints have been considered with the smooth and non-smooth objective functions. The penalizing method is used to deal with the constraints, which is required to be zero in all circumstances. The proposed method has produced

better performance for all objectives. In [165] PSO algorithm has been successfully applied to power system operation problems.

An Improved version of PSO algorithm (IPSO) has been proposed by [11]. IPSO has combined the chaotic sequence mechanism to more enhance the influence of inertia weight ω and control the movement of particle during the search process. Another technique of crossover has also been used to improve the global exploration of the algorithm. IPSO has been applied with the modified functionalities to the smooth and non-smooth ELD problems. The algorithm has been tested on the large-scale case-study and has produced promising results. In [166] has also used PSO algorithm for various ELD problems such as ELD with smooth objectives and ELD with non-smooth having multiple-fuels constraints. The effectiveness of proposed PSO method has been presented.

In [12] a brief survey of literature related to PSO algorithm and its applications in ELD problems has been presented. Later on, PSO algorithm has been implemented to solve ELD with convex and non-covex functions. Novel functionalities have been added to efficiently control the inertia weight and acceleration parameters of PSO algorithm. The best performance of the proposed work has been presented. In a very recent publication [167] an improved PSO algorithm (IPSO) has been applied to ELD problems with multi-area constraints. The case of 40 units has been adopted to validate the performance of the proposed IPSO algorithm promising results have been reported.

According to the above mentioned literature it has been concluded that ELD is a very complex, non-differentiable and highly nonlinear problem. It is also very important optimization problem in power system operations. A solution method is always required that can easy solve the problem by minimize the operating cost and satisfying all constraints. Therefore, the research in this area is always encouraged by the research community. In this thesis we have developed two variants of PSO algorithms namely NS-MJPSO Chapter 3 and NS-SPSO Chapter 4. Extensive mathematical benchmark testing have been carried out. To further validate the performance of the newly developed algorithms, we have implemented both of the algorithms to ELD problems. These algorithms are completely new, some existing objectives functions and related test data is taken from the above mentioned literature [10; 11; 15; 121]. Some other well-known variants of PSO have also been implemented for the same type of problems. The variants are Comprehensive-learning PSO (CL-PSO) [4] and Social-learning PSO (SL-PSO) [14]. In a nut-shell four new algorithms have been implemented here ELD problems. Genetic Algorithm (GA) and PSO with constriction factor have also been re-implemented and described in [10] to compare the results of new algorithms. In the next Section 5.3 we formulate the various existing problems and onwards the simulation results 5.5.

5.3 Problem Formulation

5.3.1 Smooth / Simplified Cost Function

Economic Load Dispatch (ELD) is measured as the sub-problem of Unit Commitment (UC) problems in power systems. ELD problem aims to produce the power with the minimum per unit cost, while all the systems constraints are satisfied. In the smooth or simplified ELD problems the quadratic function is applied to measure the cost of each generator. The basic constraint in the simplified cost functions are the equality and inequality constraints. The mathematical method are also able to find the solution for simplified cost functions [121].

$$min(C.F) = \sum_{j=1}^{m} F_j(P_j) = \sum_{j=1}^{m} (a_j + b_j P_j + c_j P_j^2) \tag{5.1}$$

where C represents the total generation cost F_j represents the function of generator j, a_j, b_j, c_j are the fuel cost coefficients of each generator. P_j is the generated power by unit j and m represents the total number of generating units. To dispatch the exact amount of power to the loads it is also important to calculate the possible transmission losses. That has further two categories of fixed losses and dynamic losses. While here we ignore power losses for simplicity. The first condition that needs to be satisfied in all cases is the power balance or equality constraints it means that the generated power must be equal to the demand from the loads side [10].

$$\sum_{j=1}^{m} P_{jt} = P_{load,t} \tag{5.2}$$

Secondly, the minimum and maximum limits are defined before solving the cost function all generators must be operated within the specified limits. This represents the inequality type constraints within generating units. As some generators have the large power production capacity and some have the small capacity given as follows [121]:

$$P_{j,min} \leq P_j \leq P_{j,max} \tag{5.3}$$

5.3.2 Non-smooth Cost Funtion

The involvement of valve-point loading effects in the cost objective function determines the problem to be highly non-linear and non-smooth. The simplified cost function is differentiable but the cost function including valve-point loading effects is non-differentiable. Basically, the valve-point loading effects is defined as the *superposition of sinusoidal* and quadratic functions [121]. In static ELD problems, we consider static load for a specific time period during the whole trial of the algorithm. The ELD objective function is a challenging task because of the multiple local optimum. For instance, in the six unit system we will have six local optima. We have to find the minimum and optimum setting for each unit. In the valve-point load effects the input-output curve is totally different than the smooth cost function. The cost function with valve-point loading effects is given as follows:

$$F_j(P_j) = \sum_{j=1}^{m}(a_j + b_j P_j + c_j P_j^2 + \mid e_j \times sin(f_j \times (P_{j,min} - P_j))) \tag{5.4}$$

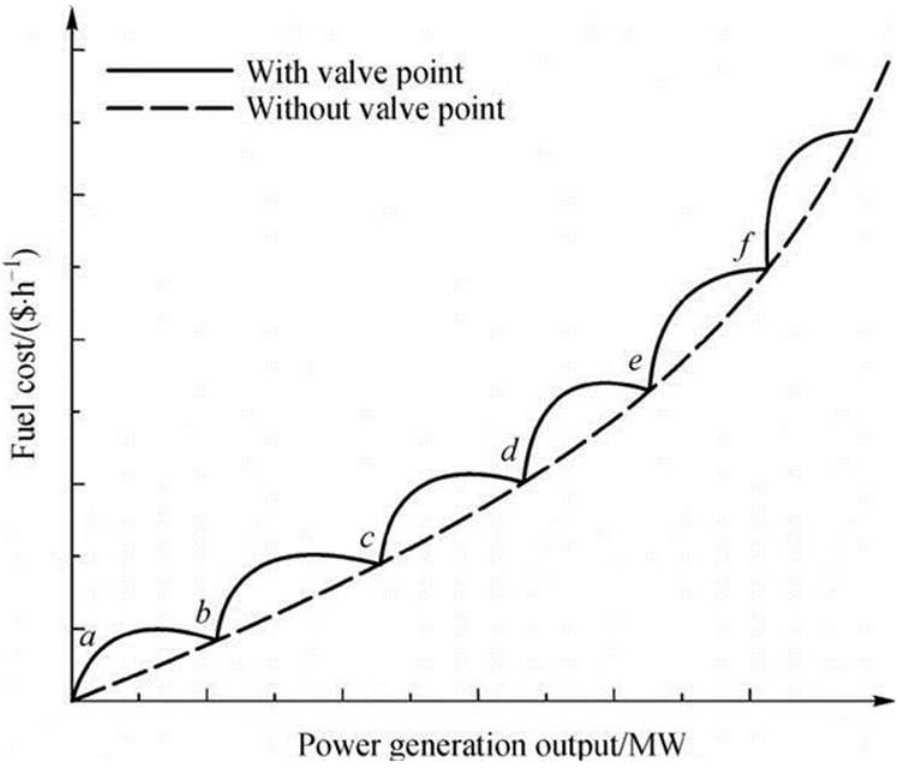


Figure 5.1: The smooth and non-smooth cost curves [121]

In the above Equation (5.4), the value of e_j and f_j represent the fuel coefficients of the

system unit j and Figure 5.1.

5.3.3 Practical Generator Constraints

(a) Power balance constraints: In Equation (5.2), we have presented the power balance and equality constraint without transmission losses. Whereas, the Equation (5.5) represent the equality constraint considering transmission losses. The total generated power must be equal to the total load in addition to the transmission losses.

$$\sum_{j=1}^{m} P_j = P_{load} + P_{loss} \tag{5.5}$$

(b) Power losses: The line losses are calculated as the using B matrix of coefficients as the unit power output [168]. The equation for losses calculation is given as follows:

$$P_{losses} = \sum_{i=1}^{m}\sum_{j=1}^{m}(P_i B_{ij} P_j + \sum_{i=1}^{m} B_{0i} P_i) + B_{00} \tag{5.6}$$

(c) Generating capacity limits: As given in the above equation (5.3) each generator should be operated according to their minimum and maximum capacity. Whereas, violating the capacity limit will cause reliability and security problem in the system.

(d) Ramp-rate limits: The operating speed of all connected units is controlled by the ramp-rate limits specified with each unit [157; 160; 161]. Equation (5.7) is applied when the generation is increasing while Equation (5.8) is applied when decreasing generation of the jth unit. The equations of ramp-rate limits are given as follows:

$$P_j - P_j^0 \leq UR_j \tag{5.7}$$

$$P_j - P_j^0 \leq DR_j \tag{5.8}$$

(e) Prohibited operating zones: Due to the valve-point effects, it is sometimes impossible to operate the generator for all ranges. Actually, this is sometimes caused by the

internal mechanical fault in the generator. There may be number of zones in which the operation should be avoided and constrained. When the unit is operated in the prohibited zone it causes the trembling and fluctuating effects in the quality of power, which is dangerous. The prohibited zones are described in the following equations:

$$P_j^m in \leq P_j \leq P_{j,1}^l \tag{5.9}$$

$$P_{j,k-1}^m in \leq P_j \leq P_{j,k}^l \tag{5.10}$$

$$P_{j,n}^u \leq P_j \leq P_j^m ax \tag{5.11}$$

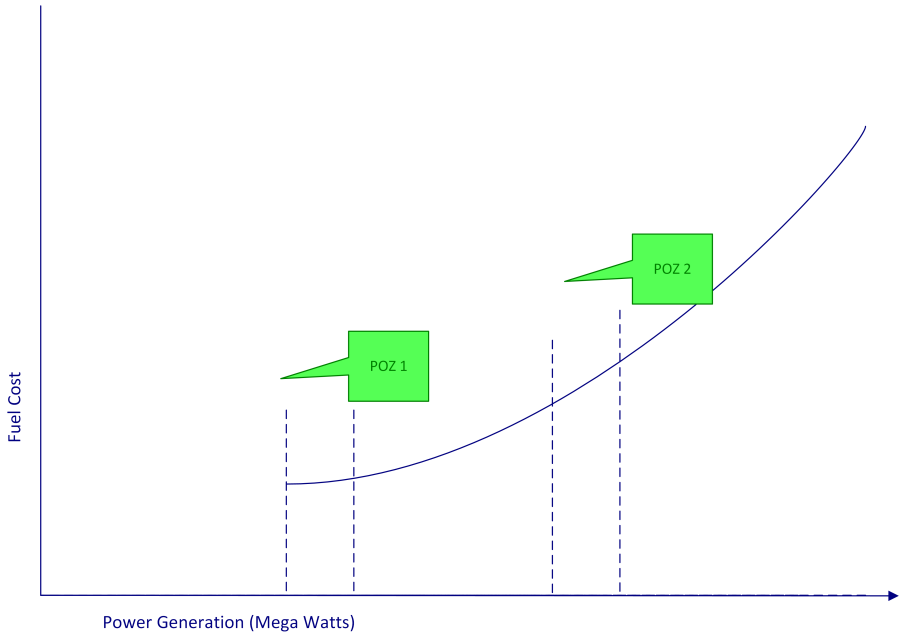


Figure 5.2: The prohibited operating zones

where, l and u are the lower and upper points of the prohibited operating zones and $k = 2, 3, ...pz_j$ represents the number of zones.

In forthcoming section 5.4, we present the implementation procedure of the newly developed NS-MJPSO and NS-SPSO for ELD smooth and non-smooth functions.

5.4 Implementation of Proposed Algorithms

The step-wise implementation procedure of newly developed NS-MJPSO and NS-SPSO is described as follows:

1. Initialize the particle's position and velocity randomly within the boundary of PC_max and PC_min. PC_max represents the maximum generation cost of all individuals, while PC_min represents the minimum cost of all individuals. A range of minimum and maximum for each individual unit is determined by using Equation 5.1 for the simplified case and Equation 5.4 for the non-smooth objective with valve-point loading effects. The decision variables P_j in the cost function of ELD problems represent the power generated by each unit. Therefore, the population of particles is initialized as follows:

$$P_{G_i} = [P_{i1}, P_{i2}, P_{i3}, \cdots, P_{iD}], i = 1, 2, \cdots, n \tag{5.12}$$

   n represents the number of particles and D is the number of units or generators. For example P_{i1} means the power generated by unit 1. In this function the dimensionality of the problem is described as [Population $\times$ Number of systems units]. The relevant constraints are considered on the case-study bases.

2. The number of N states are assigned and the resultant population is then applied an evolutionary technique. Population distribution is derived by Chapter 3, Section 3.3.1, Equation (3.8). Furthermore, by using Equation (3.9) the evolutionary factor value is described.

3. The cost function is evaluated for all particles. The evaluation criteria is $\frac{1}{f_t}$ [165]. The cost function and power generation are both correlative in evaluation function. A smaller value of evaluation is obtained from the large values of cost and power out functions.

4. *Pbest* is updated according to the current evaluation value. *gbest* is determined as the best individual in the whole population.

5. In the velocity update equation a stochastic Markov Jumping parameter is introduced. The particle jumps from its current state according the transition probability 0.9 for the initial state and then the next is predicted accordingly. A set of N acceleration coefficients are pre-defined and a specific value is selected according to the current state automatically. Later on getting all parameter values the velocity and position for all particles are then updated as the follows:

$$\begin{aligned} v_{ij}(t+1) =& \omega v_i(t) + c_1(\delta(t))rand_1(t)(pbest_{ij}(t) - P_{ij}(t)) \\ &+ c_2(\delta(t))rand_2(t)(gbest_{ij}(t) - P_{ij}(t)), \end{aligned} \tag{5.13}$$

$$P_{ij}(t+1) = P_{ij}(t) + v_{ij}(t+1) \tag{5.14}$$

$i = 1, 2, \cdots, n$, and $j = 1, 2, \cdots, m$, where n represent the number of particles and m represents the number of generating units in the system.

$P_{ij}(t+1)$ is the new updated position according to the new *Pbest* and *gbest* values. It is essential for newly updated *Pbestval* to be in the limits of its constraints. Therefore, the population is initialized according to the boundaries of constraints and no risk of violation is involved in this case.

6. *Pbest* and *gbest* are updated for all particles according to their current values of evaluations.

7. Repeat step 2 until the maximum number of iterations reach.

8. The *gbest* obtained in the last iteration is the optimum solution for the problem.

In the implementation of second newly developed NS-SPSO the only difference is the exclusion of Markov Jumping technique. The rest of the procedure is similar to the implementation of NS-MJPSO algorithm above. Furthermore, the flowchart of NS-SPSO implementation in ELD problem is given as follows:

5.5 Experimental Setup and Simulation Results

The performance of proposed algorithms is examined by choosing different types of smooth and non-smooth objective functions along-with several constraints. The case-studies used in applications of the newly developed algorithm are, six-unit system with considering transmission losses, thirteen-unit system with non-smooth cost function or valve-point loading effects. Transmission losses are ignored here. Fifteen-unit system, simplified cost function with transmission losses and non-smooth constraints, such as, two prohibited operating zones and ramp-rate limits constraints. 40-unit system cost function with valve-point loading effects. 140-unit system with valve-point loading effects and non-smooth

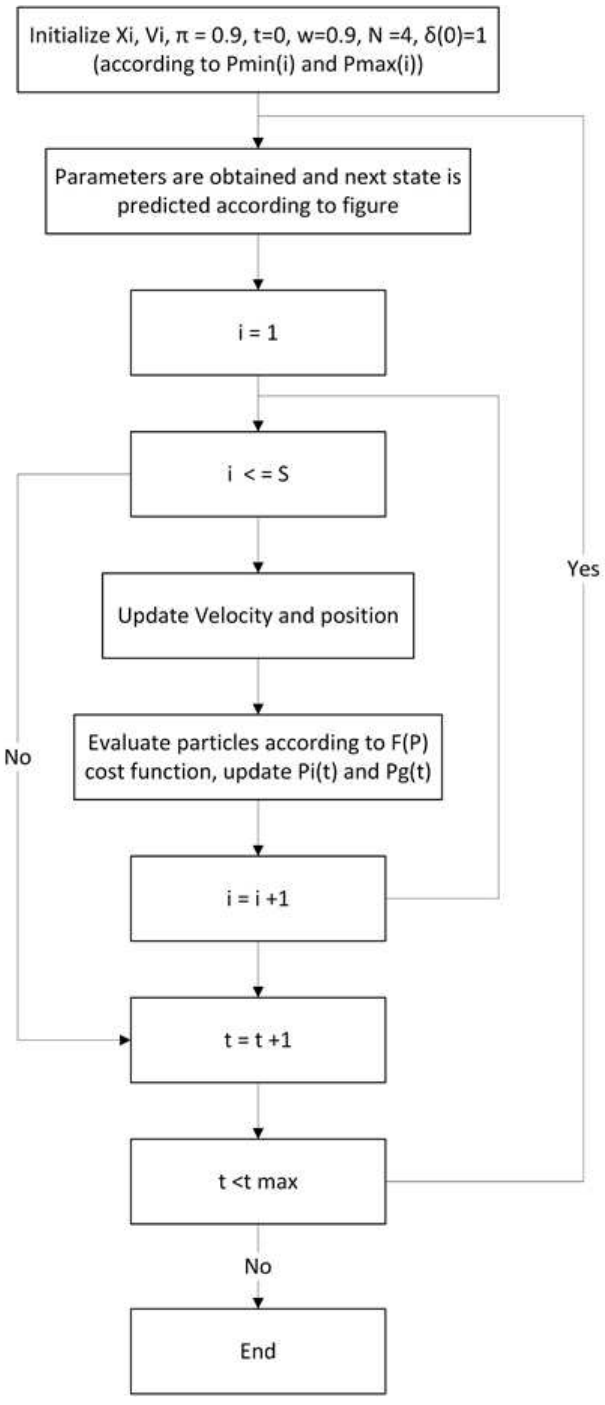


Figure 5.3: The Flowchart of NS-MJPSO Implementation to ELD problem

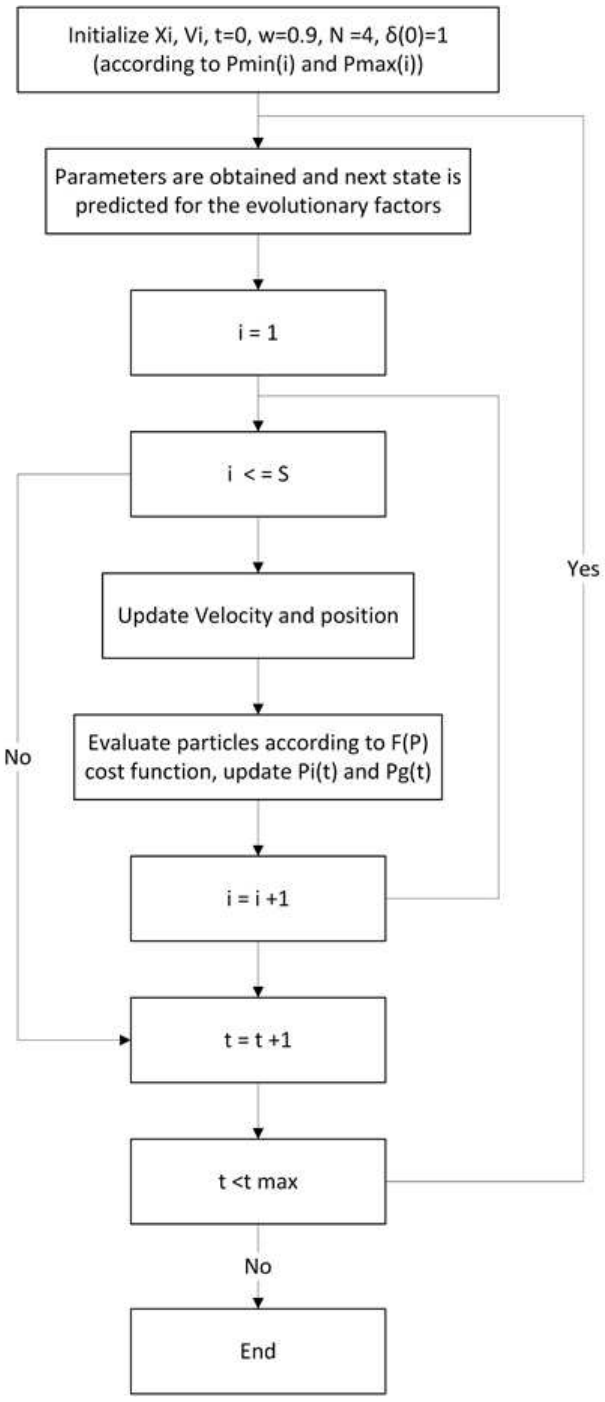


Figure 5.4: The Flowchart of NS-SPSO Implementation to ELD problem

constraints. The data used for simulating the above case-studies is taken from the published articles. All the experimental work is carried-out in Matlab R2011b, Intel Corei5 3.30GHZ 64bit Windows 7 operating system, with 4GB RAM. The proposed algorithms are compiled for 30 independent trials. The average/best results are tabulated and plotted in terms of the minimum cost and convergence time. The parameters setting for the proposed algorithms and all other variants used in the experimental work are listed in the following Table 5.1.

Table 5.1: Parameter setting of PSO for ELD applications

Algorithms	Acceleration	ω_{max}	ω_{min}	Cons; Factor (χ)	States
NS-MJPSO	C = [2, N]	0.9	0.5		N
NS-SPSO	C = [2, N]	0.9	0.5		N
MPSO	$c_1, c_2 = 2$	0.9	0.4	1	
SL-PSO	α	β=0.01			
CL-PSO	$c_1, c_2 = 1.49445$				
GA	C.R = 0.8	$P_{mut} = 0.01$	C.P = 0.5		

5.5.1 Case A: 6 Unit System

The six unit system consists of six generation units with individual capacity of power generations. Each one has its own maximum and minimum generation limit. The parameters are chosen from Table 5.1. The power load demand is assigned 1263 in the beginning. The load here is taken from literature [10] the experimental work in cited paper is repeated for comparison. Power balance, prohibited operating zones, ramp-rate limits constraints are applied on the objective. Transmission losses are taken into account. In Table 5.2, it is shown that the proposed NS-MJPSO and NS-SPSO has the average/best minimum value for the cost function. The tabular values are recorded according to the 500 iterations. In graphical illustrations all the algorithms have been tested on minimum iteration of 50. It has been observed that the proposed algorithm converge very quickly for 6 unit system. The average/best results have been drawn using individual, three and four algorithms for convergence comparisons.

5.5.2 Case B: 13 Unit System

The 13 unit system consists of 13 generation units with individual capacity of power generations. Each unit has its own maximum and minimum generation limit. The parameters are chosen from Table 5.1. The static power load demand is 1500. Power balance, equality and inequality constraints are taken into account. The non-smooth cost function with valve-point loading is considered without power losses. In Table 5.3, NS-MJPSO, NS-SPSO,

Table 5.2: Best Solutions for 6 Unit System

Unit: Load 1263 (MW)	NSMJPSO	NSSPSO	SLPSO	CLPSO	MPSO	GA
1	436.84	435.40	486.83	469.24	447.84	447.53
2	172.56	186.82	185.32	177.09	172.41	174.62
3	250.96	257.14	203.90	252.22	241.93	264.17
4	144.65	139.25	138.79	133.04	143.27	145.86
5	182.60	164.20	174.05	153.92	172.10	168.27
6	88.00	92.68	86.74	90.06	97.95	75.99
Total Power (MW)	1275.61	1275.48	1275.62	1275.57	1275.50	1276.44
Plosses	12.61	12.48	12.62	12.57	12.96	12.45
Total Cost ($):	15449.78	15447.93	15490.76	15452.98	15450.56	15459.00
No. of Hits to the Global:	30	28	24	15	17	22

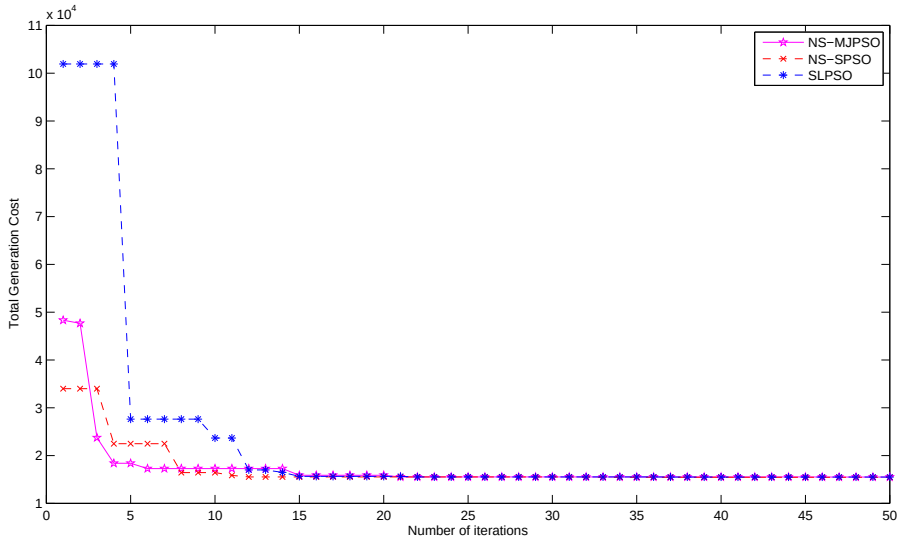


Figure 5.5: Comparison of three new algorithms for six unit system

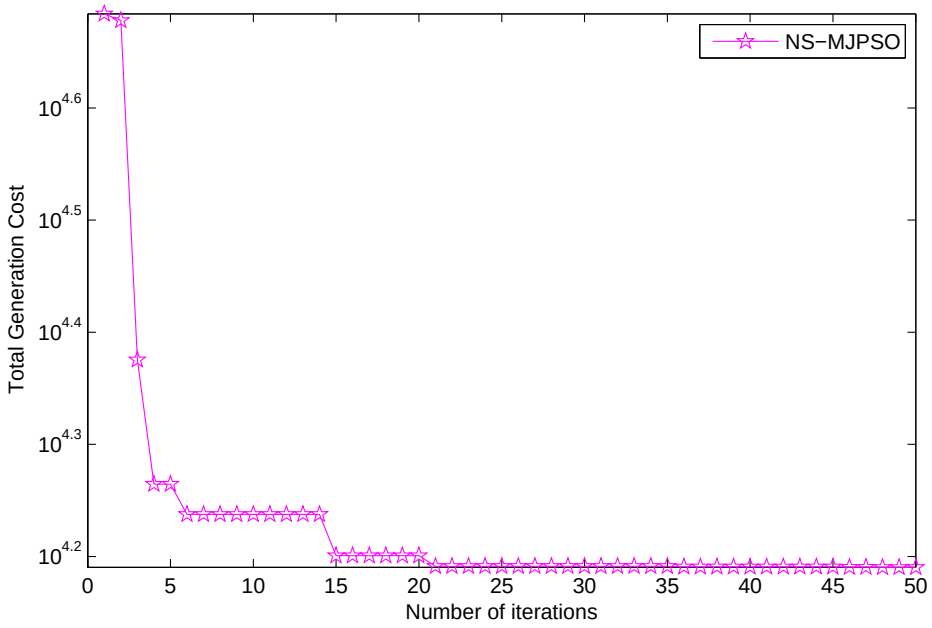


Figure 5.6: Convergence of NS-MJPSO for six unit system

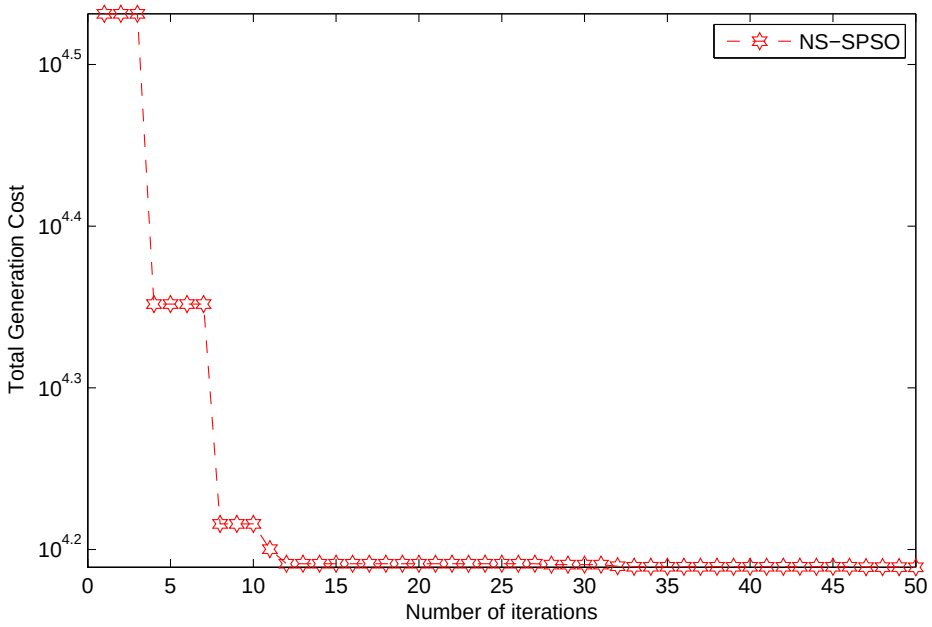


Figure 5.7: Convergence of NS-SPSO for six unit system

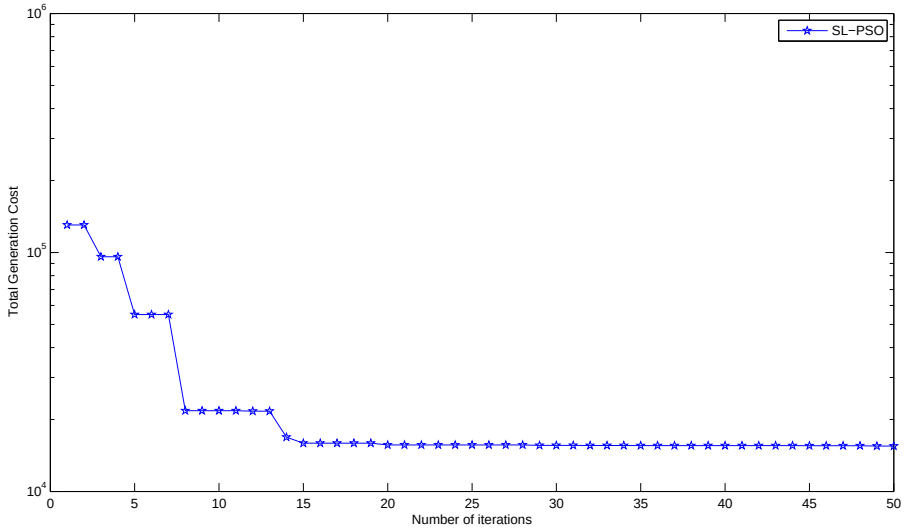


Figure 5.8: Convergence of SL-PSO for six unit system

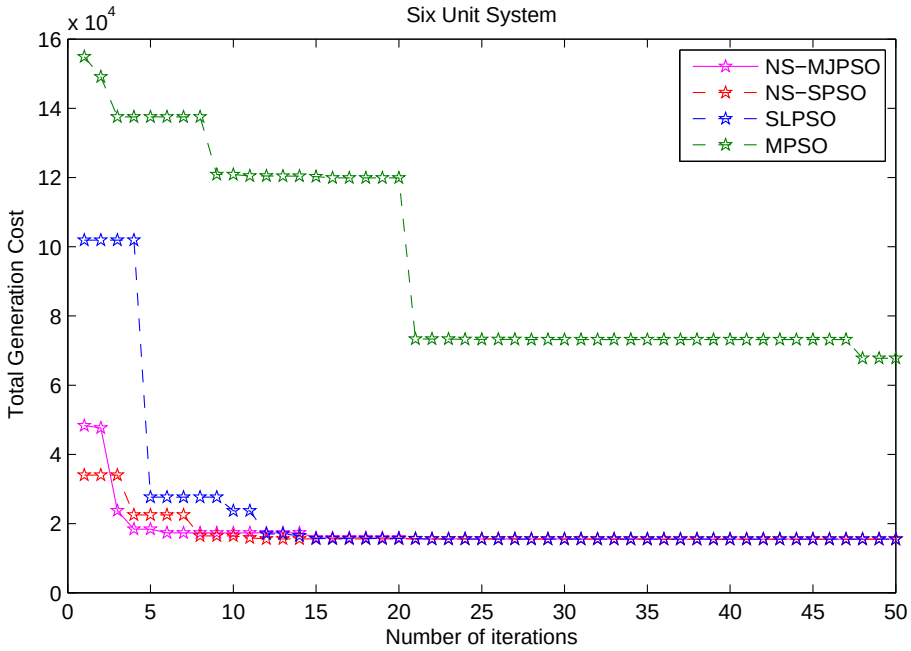


Figure 5.9: Comparison for 6 unit system

SL-PSO, and MPSO are compared. It has been observed that NS-SPSO has produced the best value for the non-smooth objective with quick convergence. With the increase in dimensionality of the large scale ELD case study, we increase the number of states N for NS-MJPSO and also adjustment other parameters accordingly. The SL-PSO does not need any parameter adjustment. It is very pleasant to mentioned that the performance of SL-PSO produce better results when the dimensionality is increased.

Table 5.3: Best Solutions for 13 Unit System

Unit Number: Load 1500 (MW)	NSMJPSO	NSSPSO	SLPSO	MPSO
1	95.79286598	182.2398811	107.6585472	363.112936
2	316.8700731	142.8536507	303.5117626	35.3225141
3	119.0853497	67.27843621	85.55723605	64.9611761
4	74.83152494	167.4491739	158.98175	93.0367954
5	132.0549546	159.5565155	132.5964763	92.6804463
6	111.4418788	115.1790597	65.89578758	106.44491
7	91.21800746	108.4766563	104.1599714	157.293324
8	93.98403194	104.1208493	122.2233338	115.055225
9	115.9471883	129.7844851	61.9532563	155.369062
10	87.17892853	81.58783452	108.1172993	57.2371455
11	70.89015813	61.06157753	101.3004781	83.015422
12	88.7020085	91.57029704	83.34279265	92.6105774
13	102.0030343	88.84154297	64.69932615	83.8606284
Total Power (MW):	1500.000004	1499.99996	1499.998017	1500.00016
Total Cost ($):	16507.91229	16227.5131	16579.41014	16590.2921
No. of Hits to the Global:	30	27	28	20

5.5.3 Case C: 15 Unit System

In the 15 unit system, we have 15 generation units ready to dispatch a scheduled load to the consumer. Similar to the other systems mentioned earlier each unit of the system has its own characteristics and limitations with it. The power generation capacity and fuel type. The power load demand is assigned 1263. The simulation trial consist of NS-MJPSO, NS-SPSO, SL-PSO and MPSO. Parameters used here are given in Table 5.1. The load here is taken from literature [10], the experimental work in cited paper is repeated for comparison. Power balance, prohibited operating zones, ramp-rate limits constraints are applied on the objective. Transmission losses are taken into account and the smooth cost function is applied. In Table 5.4 the best value of the proposed NS-MJPSO and NS-SPSO is presented. SL-PSO has also produced similar best performance. However, the MPSO

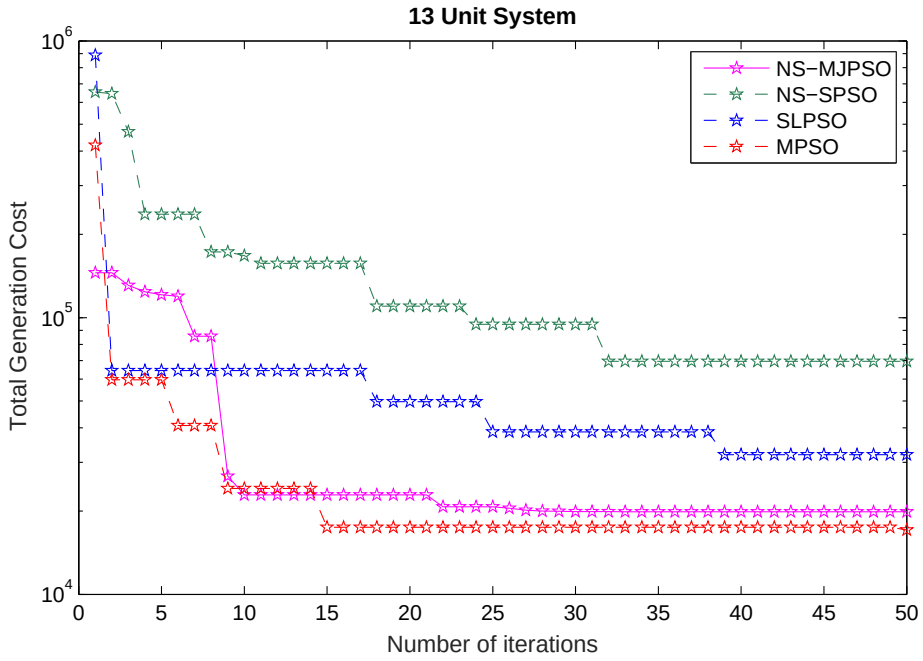


Figure 5.10: Comparison for 13 unit system

results in the violation of power balance constraint and slow convergence is observed for MPSO.

5.5.4 Case D: 40 Unit System

In this section a large-scale case system having 40 units is considered for ELD optimization. Similar to the other systems, in 40 unit system each unit has minimum maximum generation capacity. The fuel type and the quantity consume in a specific time is different for each unit. Further, to check the performance of proposed methods the non-smooth objective is applied. The cost function with valve-point loading effect is selected for simulations. Parameter setting for participant algorithms is given in Table 5.1. A static load demand of 9500 is assigned to the system. Power balance and capacity limit constraints are applied to the objective with valve-point effects. The transmission losses are not taken into account. The comparative numerical results for the proposed algorithms NS-MJPSO, NS-SPSO, SL-PSO, and MPSO are given in Table 5.5 the best performance value of the proposed NS-MJPSO is attained by adjusting the state parameter N to 8 for Case D. SL-PSO algorithm has also shown the best performance without any further adjustments in the parameters.

Table 5.4: Best Solutions for 15 Unit System

Unit Number: Load 2630 (MW)	NSMJPSO	NSSPSO	SLPSO	MPSO
1	424.36393	454.0291967	442.877078	357.6578001
2	370.9496103	353.815462	377.0892621	379.8616996
3	117.7955535	98.79280078	124.202704	89.31085749
4	106.1453568	123.9648658	90.34432079	95.36506944
5	167.3174254	165.550134	164.6204538	169.9825838
6	408.7033798	457.8002511	421.8109257	364.9406572
7	428.137482	370.0860964	405.7983	429.9920737
8	150.9145779	153.6917196	150.0174873	159.7795769
9	136.1522176	155.5497163	118.8825342	99.10497971
10	141.8635466	87.51145252	134.1305135	143.9408787
11	63.35567612	64.82467187	58.55379154	79.96617673
12	40.57532433	45.81854962	53.24457165	79.81938258
13	38.21741124	58.815408	57.25296578	84.97916389
14	30.40783776	52.22544283	30.50418412	54.79302336
15	44.63332728	24.18195852	37.13601629	54.85357494
Total Power(MW):	2669.532657	2666.657726	2666.465109	2644.347498
Plosses:	39.5326	36.6578	36.4674	37.015
Total Cost ($):	33068.89888	33028.16645	33097.69123	33080.24547
No. of Hits to the Global:	29	26	23	19

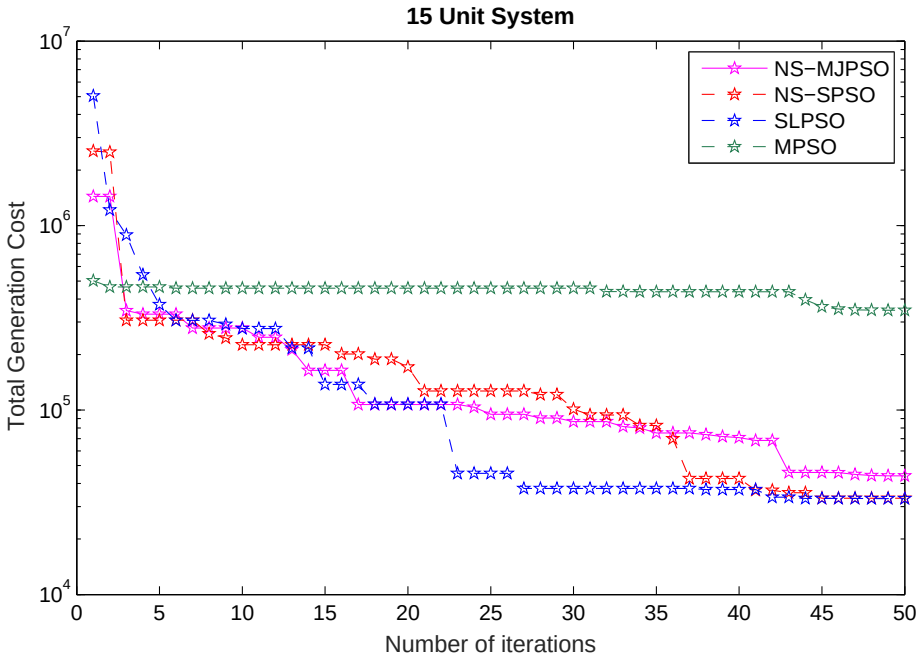


Figure 5.11: Comparison for 15 unit system

Table 5.5: Best Solutions for 40 Unit System

Unit No: Load 9500 (MW)	NSMJPSO	NSSPSO	SLPSO	MPSO
1	105.2763023	54.36141766	89.65049281	55.60312634
2	106.3272433	114.9336624	88.35001797	96.79075246
3	101.8853216	110.7510351	91.18266616	86.57523343
4	146.4207168	133.7184443	147.0730032	109.7965506
5	72.28923529	30.83815624	71.9111531	91.94879312
6	113.5372156	130.5228112	105.4113329	139.1210506
7	212.6523657	220.4903892	123.5157097	156.1350066
8	236.3301774	241.9206245	262.3924933	238.5438842
9	175.4936283	258.5024445	105.585812	221.4013641
10	255.0672382	167.2667958	210.6693654	188.9100567
11	241.2235223	261.7387391	232.3202098	325.3765072
12	146.943214	347.6314891	278.1612274	343.3659489
13	412.0492832	318.8406414	209.8925082	316.5831051
14	479.4330058	197.1559632	523.5517469	400.3203505
15	463.3283763	339.320376	246.0998479	243.243218
16	273.1399991	278.3131924	312.2950008	468.4193495
17	330.8347718	382.0060741	411.9934604	306.3856416
18	445.0723037	518.2093144	406.7828501	364.5669838
19	335.7462825	354.2196252	381.2570696	434.6341982
20	471.7224498	373.6855338	516.1914945	395.2450412
21	329.5018117	487.4487746	524.3624865	462.5189282
22	444.6256026	392.1304961	493.4815864	470.2142561
23	388.2858889	584.0682993	379.5160824	296.576965
24	417.6786353	386.8222065	446.4244823	468.2384674
25	383.6836977	539.6608024	498.7724283	476.1219453
26	396.8553993	334.244159	334.9644342	466.1826276
27	33.86628426	138.293121	46.42394982	118.7365872
28	134.2877497	95.30784823	117.8705716	120.2070402
29	61.72047685	-0.809802387	113.0544898	82.65821346
30	97.37006223	128.9600513	69.96853767	68.93778027
31	173.7900058	106.7782538	200.7194844	112.3820918
32	166.7008562	149.7104347	152.864569	71.40280654
33	127.8047764	49.04857412	94.9008432	189.9070338
34	128.1455848	128.4972193	130.0616722	170.7904678
35	144.2376887	167.5767652	152.109724	123.822739
36	141.0150155	134.4474218	155.6138718	175.7119654
37	96.89331476	80.68636174	36.45839472	102.3998883
38	82.27939162	85.59712427	70.73042044	100.7943437
39	55.16845168	112.9105634	94.37739063	62.29715242
40	571.3167124	564.1945561	572.5411988	377.1325409
Total Power(MW):	9500.000059	9499.99996	9499.50408	9500.000003
Total Cost($):	130437.6804	132347.9273	131601.8147	136512.9625
No. of Hits to the Global:	30	28	27	21

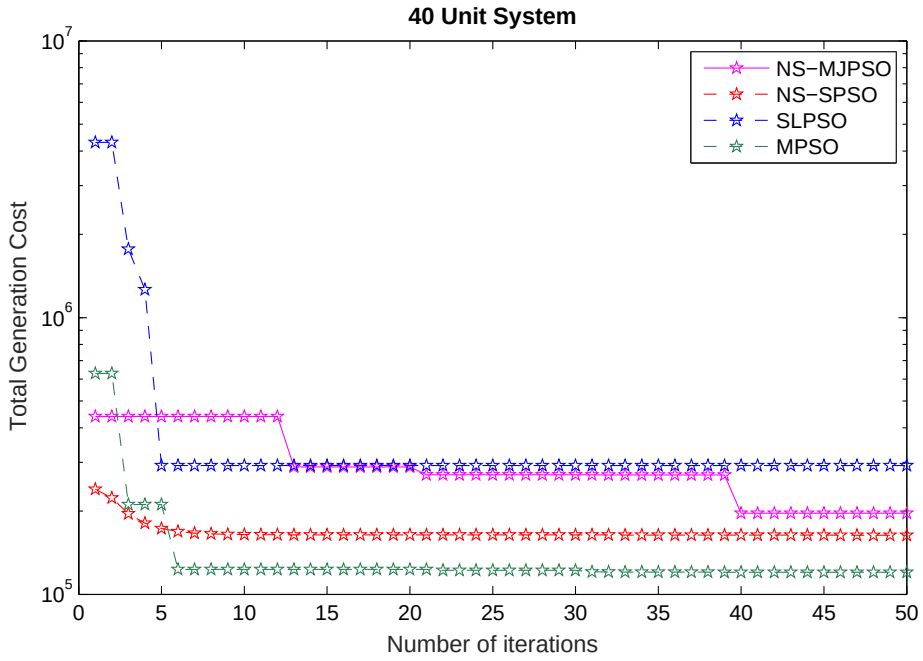


Figure 5.12: Comparison for 40 unit system

5.5.5 Case E: 140 Unit System

In this section a very large-scale power system having 140 units is selected. Similar to the other case studies, each unit has minimum maximum generation capacity. The fuel type and the fuel quantity consumed in a specific time differs for each unit. Henceforth, to calculate the minimum cost for such a large-scale power system, the cost function with valve-point loading effect is selected for simulations. Parameter setting for participant algorithms is given in Table 5.1. A static load demand of 40300 is assigned to the system. Power balance, capacity limit, prohibited operating zones and ramp-rate limits constraints are applied to the objective with valve-point effects. The transmission losses are not taken into account. Due to the extremely large scale study, the performance of NS-MJPSO and NS-SPSO is unstable. Therefore, we ignore the results produced by the proposed algorithms for this system. However, a newly developed SL-PSO is applied here. The best result produced for this case study in terms of minimum cost and shortest computation time for SL-PSO are given in Table 5.6 and the individual results of SL-PSO is also illustrated in the following Figure 5.13.

Table 5.6: SL-PSO application in 140 Unit system

SL-PSO	**Load: 40300MW**										
Unit	**Power(MW)**	**Cost($)**	**Unit**	**Power(MW)**	**Cost($)**	**Unit**	**Power(MW)**	**Cost($)**	**Unit**	**Power(MW)**	**Cost($)**
1	106.3064272	8102.732357	36	285.0850769	5439.116	71	189.3874209	17459.86	106	822.9381303	2585.4259
2	130.4682043	6817.65062	37	147.0341701	2904.849	72	174.9194906	18710.52	107	822.1824487	2661.8001
3	143.4745052	7597.823745	38	141.6777152	2811.681	73	236.5526015	17804.59	108	858.4983643	2955.2097
4	140.8891301	7475.265291	39	428.5571003	8549.729	74	238.2972573	18071.73	109	818.0271658	2851.3838
5	132.6004792	9915.694239	40	450.3218768	8878.702	75	228.6611969	17387.77	110	813.1462769	2694.8031
6	122.9443232	9090.732419	41	7.995005235	1361.002	76	213.9848218	16407.4	111	828.2801694	3043.7575
7	334.6401009	6552.295786	42	22.69600339	3131.489	77	279.8830949	20964	112	139.6477469	14093.633
8	300.2106396	5931.148451	43	215.129273	16740.58	78	352.0874189	27163.52	113	114.3312484	11727.151
9	341.3521402	6425.306665	44	191.6632019	15259.06	79	229.7994213	9505.859	114	189.3189539	18790.161
10	389.4170115	7238.847111	45	204.7229737	16158.33	80	204.0741278	8558.099	115	285.6488397	25751.918
11	306.5397423	5850.683597	46	169.8179526	13711.11	81	297.0261489	22618.7	116	263.9373866	23998.528
12	336.1759195	6381.844093	47	191.838857	15062.98	82	90.73864168	10179.1	117	270.4512179	24521.58
13	305.4463903	6015.552602	48	178.3518984	14350.81	83	173.019782	18331.18	118	104.8177393	10486.834
14	381.3856455	7331.605088	49	216.1355331	16893.68	84	131.0066572	14333.43	119	127.4516076	12742.225
15	292.5349754	5796.268678	50	206.3234466	16286.4	85	163.1292488	17404.24	120	131.2324344	12710.252
16	278.7282698	5563.218418	51	216.8140654	17420.9	86	214.1265928	23954.98	121	214.3405187	20274.296
17	319.7694639	6272.563986	52	235.9497167	18791.09	87	266.0234949	29348.29	122	9.679993326	1242.7654
18	290.9491558	5778.859149	53	272.4875642	21407.43	88	285.4455397	24131.91	123	43.28525657	6095.0053
19	317.4251798	6252.50965	54	213.8011546	17205.17	89	239.4686209	20662.51	124	47.68719974	5952.9208
20	302.3612007	5997.336495	55	194.4194815	19117.93	90	240.5571466	20592.4	125	31.12943153	4034.651
21	269.6812448	5449.509179	56	185.8756783	19084.18	91	199.1672546	17681.18	126	22.69641832	3107.2287
22	287.1378884	5741.162939	57	177.5811607	18190.16	92	542.8415537	1788.414	127	11.51001947	1822.9028
23	285.4491007	5698.33185	58	232.3390069	22091.22	93	515.2107672	1967.634	128	130.6622738	13361.17
24	325.5791952	6391.330361	59	168.9087685	16422.27	94	815.6556347	2696.893	129	19.87954784	2646.6346
25	363.8227756	7106.350347	60	399.7066936	34902.14	95	797.1593412	2591.829	130	25.81708567	3110.6754
26	345.1572333	6750.830946	61	230.7725147	21816.85	96	642.6320654	1514.926	131	7.583154762	1070.2056
27	462.6566133	8813.623503	62	150.3075408	13977.36	97	647.8504883	1490.316	132	77.38882207	7385.0689
28	313.0470969	6124.650704	63	189.3540505	16390.51	98	662.9955471	1625.911	133	2.184185275	479.53046
29	404.0164382	7486.146092	64	203.7028865	17397.13	99	630.6121107	1581.625	134	53.82287424	6018.2775
30	287.2842633	5490.910849	65	214.5425766	18654.86	100	885.0191434	2934.108	135	47.25816378	5336.0456
31	305.2790406	5994.167544	66	217.167896	18108.18	101	905.9780517	3003.896	136	58.99052487	10908.959
32	371.886665	7155.239743	67	328.5301844	25501.47	102	878.0288656	2702.312	137	27.06282749	8860.5017
33	310.1960199	6078.803571	68	234.1009531	19794.86	103	873.093209	2835.928	138	16.2108959	1496.9667
34	282.1412967	5598.196459	69	144.4333727	14609.04	104	921.8730302	2935.53	139	8.67895188	812.94704
35	289.1630468	5586.312369	70	179.3800683	20551.89	105	842.7744356	2556.324	140	25.23894645	2372.7271
Total:										**40303.73929**	**1482328.7**

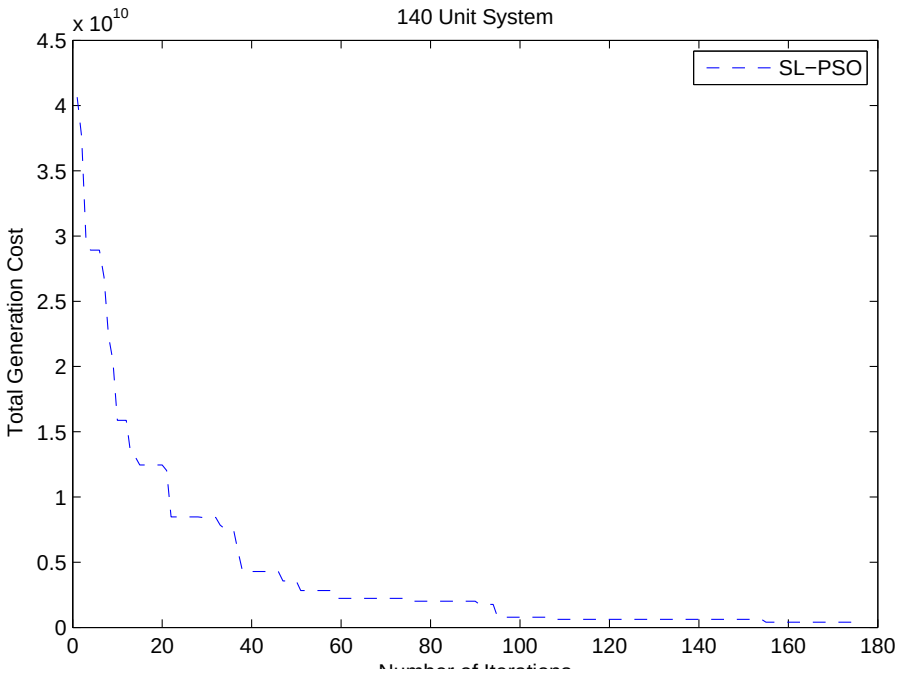


Figure 5.13: SL-PSO performance for 140 unit system

5.6 Summary of Applications

In this chapter, we have introduced three algorithms for the applications in ELD problems. Two of them namely N State Markov Jumping Particle Swarm Optimization (NS-MJPSO) and N State Switching Particle Swarm Optimization (NS-SPSO) are two brand new algorithms developed in this thesis. NS-MJPSO is a stochastic search technique and NS-SPSO combines an evolutionary concept with traditional PSO. While the third one is Social-learning (SL-PSO) taken from the recently published article [14] developed for high-dimensional problems. In the mentioned paper, SL-PSO has been tested for up to thousand dimension. Therefore, we have contributed an application of the SL-PSO in this chapter. Five different case studies are considered here. The data and objectives used here are downloaded from the http://www.ntu.edu.sg/home/epnsugan.

We have summarised that in the practical simulations environment, especially in Case A, Case B, Case C and Case D the proposed algorithms NS-MJPSO and NS-SPSO has produced promising results. The SL-PSO has performed very well in the large-scale and high-dimensional problems. Therefore, we conclude here that the proposed NS-MJPSO, NS-SPSO has the capability to perform well in the real-world environment of different types.

Chapter 6

Dynamic Economic Dispatch Using Novel Particle Swarm Optimization Algorithms

6.1 Introduction

In this chapter, the newly developed algorithms N State Markov Jumping Particle Swarm Optimization (NS-MJPSO) and N State Switching Particle Swarm Optimization (NS-SPSO) are proposed for the Dynamic Economic Load Dispatch (DELD). DELD determines the core objective of power system operation. A DELD describes the optimal combination of generating power with all constraints at the minimum cost, while meeting the scheduled load demand for the whole day. Time base power load demand is predicted for dispatch. Basically, DELD is the complex form of the static Economic Load Dispatch (ELD). As in static ELD, a fixed value of load is followed for the whole period. However, DELD is a high dimension problem. For instance, if we consider the 24 hours dispatch we will have 24 different loads for dispatch. Hence, the DELD problem is 24 times higher than ELD in dimensionality. Technically, the power systems units are committed to dispatch power by the dynamic variation in demand with time. The settings of each unit are adjusted for each load in each hour (time). It is a challenging task to dispatch power dynamically, control and operate the system according to increase and decrease in the load demand within the practical constraints of generating units.

Here we propose new algorithms to solve this challenging problem and obtain optimal best results in comparison to the others already applied. The performance of the new algorithm is tested by applying it to two case studies of DELD. First one is Case Study A, which represents 5 unit system with prohibited operating zones, ramp-rate limits and considering power losses as well. The main objective is non-smooth and cost function with valve-point effects. Second, is the Case Study B, that represents the 10 unit system having non-smooth objective function with valve-point loading effects. Power losses and prohibited operating zones are ignored in this case study. In the experimental work we illustrate the best performance of the newly developed algorithms in comparison to the existing methods presented in the literature. In the remaining part of the chapter, we organise the work section-wise. In Section 6.2, some background work about applications methods for DELD is summarised. Section 6.3 the problem formulation of DELD is described. Section 6.4 implementation procedure of the proposed methods is demonstrated. In Section 6.5 simulations and experimental work is presented. In the last Section 6.6 we summarise the whole work in the chapter.

6.2 Related Work

The problem of Dynamic Economic Load Dispatch (DELD) is a non-smooth problem, which represents the discretization of system. DELD is solved continuously in discrete number of intervals or states. Due to the non-linear appearances of the system structure such as prohibited operating zones constraint on generation unit and also the ramp-rate limits for variation of load. The conventional methods are very infeasible and inefficient due to this non-linearity and huge dimensionality.

Subsequently, to find a suitable and convenient approach for solving DELD problems enormous efforts have been made. Artificial intelligence methods have been tested and evaluated. Especially, Genetic Algorithm (GA) in [160], Tabu Search (TS) algorithm [169], Simulated Annealing (SA) [17], Evolutionary Programming (EP) [170] have been widely used. However, GA has been applied commonly because of the efficient technique of parallel type search. In the new multi-modal real-world problems, the weak performance of GA has been reported by several researchers. The main disadvantage of GA that has been observed in many cases is the premature convergence, which has negatively effected its reputation.

Furthermore, GA has been thoroughly and critically studied. In [17] has proposed simulating annealing (SA) for DELD problems. SA is also a global optimization technique. To solve the main objective of DELD which is a non-smooth function with non-linear type constraints, ramp-rate limits, prohibited operating zones, power balance have been applied. The transmission losses have also been calculated. SA method performance has been evaluated on a 5 unit system for the period of 24 hours. The only problem that has been magnified by the authors is computation time required. However, the overall performance in terms of the best evaluation of the cost function was good.

PSO algorithm [36] has been introduced as the successor of GA. The promising characteristics of PSO algorithm has largely diverted the attention of researchers. In [15] Particle Swarm Optimization (PSO) is used to solve the constrained DELD problem. Generator constraints such as ramp-rate limit and prohibited operating zones were applied on the objective function. Transmission losses have also been considered. Actually, this is the extension of the author's previous work [10] in which the traditional or static ELD has been solved by using Particle Swarm Optimization (PSO) algorithm. The performance has been compared for two cases of 6 and 15 units in terms of best values with other conventional and stochastic methods. PSO has performed well in comparison to other algorithms for the same objective.

In [18] an adaptive PSO (APSO) has been proposed for ELD and DELD problems.

According to APSO, an adaptive procedure for inertia weight ω has been introduced. A ranking strategy is defined for each particle and the top rank is assigned to the best particle, where the inertia ω is very small. While in contrary, the particle with the worst fitness takes the larger value for its inertia ω. Various case studies for ELD and DELD have been selected for evaluating the performance of proposed APSO. Smooth and non-smooth objective with non-linear constraints have been used. The results of proposed APSO method have been compared with the existing method. The best performance in terms of the best solutions have been presented by proposed APSO for the said objectives.

A hybrid swarm algorithm [171] has been proposed for solving DELD problem. A combination of Harmony Search (HS) and PSO has been developed as a hybrid technique for solving DELD non-smooth objectives. These type of methods are normally developed to help an existing method working efficiently in complex and dynamic environment. In three different cases the performance of proposed hybrid algorithm has been described. The non-smooth constraints such as, ramp-rate limits and power balance are considered here with transmission losses as well. It has been concluded by analysing the results that proposed hybrid method is capable of producing the best solution in such dynamic environment.

Recently, in [16] a new variant of PSO algorithm named Lbest-PSO has been used for solving DELD problems. All the system and generator constraints have been considered with objective function of the problem. The Lbest-PSO applied here has been developed in [172] it has been described by solving some case studies that Lbest with neighbourhood topology has shown the best performance in complex and multi-modal problems.

According to the above mentioned studies, it has been concluded that DELD is highly complex, non-linear and non-smooth problem. Solving the optimization problem with a static objective function is relatively simple. On the other hand, the dynamic problems have the objective that changes dynamically with time is very challenging task for all classical and heuristic methods. As mentioned above, many algorithms have been modified and applied for solving such dynamic type problems. However, there is still a need for improvement. This chapter presents an application of the newly developed NS-MJPSO and NS-SPSO in DELD problems. These algorithms have been developed and briefly evaluated for some mathematical benchmark functions. In the previous Chapter 5 both of the new algorithms have been applied to the static ELD problems for various case studies. The promising results in the static environment has encouraged to apply the same in dynamic environment DELD problems. We will consider the sample case study 5 unit system with non-smooth cost function considering ramp-rate limit, prohibited operating zones and also considering transmission losses.

6.3 Problem Formulation

Dynamic Economic Load Dispatch (DELD) is measured as the sub-problem of Unit Commitment (UC) problems in power systems. DELD problem aims to produce the power with minimum per unit cost, while all the systems constraints are fulfilled for the whole period of dispatch. In the smooth or simplified DELD problems, the quadratic function is used to measure the cost of each generator. The basic constraint in the simplified cost functions are the equality and inequality constraints. The classical methods are unable to find the solution for the non-smooth cost functions of DELD [15].

$$min(C.F)_t = \sum_{t=1}^{T}\sum_{j=1}^{m} F_j(P_j) = \sum_{t=1}^{T}\sum_{j=1}^{m}(a_j + b_j P_{jt} + c_j P_{jt}^2) \tag{6.1}$$

where C represents the total generation cost F_{jt} represents the function of generator in t time j, a_j, b_j, c_j are the fuel cost coefficients of each generator. P_{jt} is the generated power in t time by unit j, T is the total number of hours and m represents the total number of generating units. To dispatch the exact amount of power to the loads, it is also important to calculate the possible transmission losses. That has further two categories of fixed losses and dynamic losses. While here we ignore power losses for simplicity. The first condition that needs to be satisfied in all cases is the power balance or equality constraints It means that the generated power must be equal to the demand from the loads side [15].

$$\sum_{j=1}^{m} P_{jt} = P_{load,t} \tag{6.2}$$

Furthermore, the minimum and maximum limits are defined before solving the cost function and all the generators must be operated within the specified limits. This represents the inequality type constraints within generating units. As some generators have the large power production capacity and some have the small capacity given as follows: [121]:

$$P_{j,min} \leq P_{jt} \leq P_{j,max} \tag{6.3}$$

6.3.1 Cost Funtion with Valve-point Loading

The engagement of valve-point loading effects in the cost objective function determines the problem as a highly non-linear and non-smooth. The simplified cost function is differentiable, but the cost function including valve-point loading effects is non-differentiable. Basically, the valve-point loading effects is defined as the *superposition of sinusoidal* and quadratic functions [121]. In static DELD problems, we consider different load for each time period during the whole trial of the algorithm. The DELD objective function is a challenging task because of the multiple local optimum and time varying constraints. We have to find the minimum and optimum setting for each unit. In the valve-point load effects, the input-output curve is totally non-identical to the non-smooth cost function. The cost function with valve-point loading effects is given as follows:

$$F_{jt}(P_{jt}) = \sum_{t=1}^{T}\sum_{j=1}^{m}(a_j + b_j P_{jt} + c_j P_{jt}^2 + |\, e_j \times sin(f_j \times (P_{j,min} - P_{jt}))) \tag{6.4}$$

In the above Equation (6.4), the value of e_j and f_j represent the fuel coefficients of the system unit j and Figure 5.1.

6.3.2 Practical Generator Constraints

(a) Power balance constraints: In Equation (6.2), we have presented the power balance and equality constraint without transmission losses. While the Equation (6.5) represents the equality constraint with consideration of transmission losses. The total generated power must be equal to the total load in addition to the transmission losses.

$$\sum_{j=1}^{m} P_{jt} = P_{load,t} + P_{loss,t} \tag{6.5}$$

(b) Power losses: The line losses are calculated as the using B matrix of coefficients as the unit power output [168]. The equation for losses calculation is given as follows:

$$P_{losses,t} = \sum_{i=1}^{m}\sum_{j=1}^{m}(P_{i,t} B_{ij,t} P_{j,t} + \sum_{i=1}^{m} B_{0i} P_{i,t}) + B_{00,t} \tag{6.6}$$

(c) Generating capacity limits: As given in the above Equation (6.3) each generator should be operated according to their minimum and maximum capacity and violating the capacity limit will cause reliability and security problem in the system.

(d) Ramp-rate limits: The operating speed of all connected units is controlled by the ramp-rate limits specified with each unit [15; 17; 18]. Equation (6.7) is applied when the generation is increasing, while Equation (6.8) is applied when decreasing generation of the jth unit. The equations of ramp-rate limits are given as follows:

$$P_j - P_j^{t-1} \leq UR_j \tag{6.7}$$

$$P_j^{t-1} - P_j \leq DR_j \tag{6.8}$$

$$max(P_j^{min}, P_{j,t}^0 - DR_i) \leq P_{j,t} \leq min(P_j^{max}, P_{j,t}^0 + UR_i) \tag{6.9}$$

(e) Prohibited operating zones: Due to the valve-point effects, it is sometimes impossible to operate the generator for all ranges. Actually, this is because of the internal fault in the generator. There may be number of zones in which the operation in a certain range is restricted. When the unit is operated in the prohibited zone, it causes the trembling and fluctuating effects in the quality of power, which may cause problem to the system. The following equations describe prohibited zones and their minimum maximum point:

$$\begin{cases} P_j^m in \leq P_{j,t} \leq P_{j,1}^l \\ P_{j,t} \in P_{j,k-1}^m in \leq P_{j,t} \leq P_{j,k}^l \\ P_{j,n}^u \leq P_{j,t} \leq P_j^m ax \end{cases} \tag{6.10}$$

where l and u are the lower and upper points of the prohibited operating zones and $k = 2, 3, ...pz_j$ represents the number of zones.

In the following Section 6.4 we demonstrate the implementation process of the newly developed NS-MJPSO and NS-SPSO for DELD cost functions.

6.4 Implementation of the Proposed NS-MJPSO and NS-SPSO

The implementation of the proposed NS-MJPSO and NS-SPSO are quite similar. The only difference is the auxiliary parameter in the velocity update equation, which is based on Markovian Jumping in NS-MJPSO and in NS-SPSO velocity update is based on Evolutionary factor.

Here a particle represents the power output of generation unit; each particle represents a candidate solution at time t in dynamic environment. For instance, in the case of m unit system the power is dispatched to the loads, where $P_{gi,t}$ represents the ith particle described as bellow:

$$P_{gi,t} = [P_{gi,1}, P_{gi,2}, P_{gi,3}, \cdots, P_{gi,m}]_t, i = 1, 2, 3, \cdots, n \tag{6.11}$$

where $P_{gi,m}$ is the power output of ith particle, m represents the number of units and n represents population of particles in the time interval t. The population comprises dimensionality $n \times m$.

Step 1: In case of N State Markovian Jumping Particle Swarm Optimization (NS-MJPSO) the number of N states are pre-defined and the resultant population is then applied an evolutionary technique. Population distribution is derived by Chapter 3, Section 3.3.1, Equation (3.8). Furthermore, by using Equation (3.9) the evolutionary factor value is described.

Step 2: The required data for the objective function is initialized in this step for each time interval t. Such as the minimum and maximum power generating boundary for each unit is defined. The F_{Max} and F_{Min} are calculated. The population is randomly generated in the boundary of F_{Min} and F_{Max}. The constraints are applied to the initial population and all the particles have feasible values.

Step 3: Calculate transmission losses by the given Equation (6.6), $P_{loss,t}$, for each unit at time t.

Step 4: The following equation determines the evaluation function of DELD cost functions Basically, the evaluation function is the sum of total cost of generated power produced by all units plus the penalty factors λ_1 for power balance λ_2 for ramp-rate limit constraints, λ_3 for prohibited operating zones [171]. The violation of constraints is described by penalty factors λ added to the sum of total generation cost.

$$\begin{aligned} F_k = &\sum_{t=1}^{T}\sum_{i=1}^{m} F_{i,t}(P_{i,t}) + \lambda_1 \sum_{t=1}^{T}\sum_{i=1}^{m}(P_{i,t} - P_{D,t})^2 + \lambda_2 \sum_{t=1}^{T}\sum_{i=1}^{m}(P_{i,t} - P_{ramp})^2 \\ &+ \lambda_3 \sum_{t=1}^{T}\sum_{i=1}^{m}(P_{i,t} - P_{zone})^2 \end{aligned} \tag{6.12}$$

The fitness is evaluated for each unit in each time interval of dispatch. The output produced by each unit is compared with power demand in that specific time or hour. The constraint violation is also analysed here in each time interval. A large penalty value is added to the cost function in case of violation of any type constraint.

Step 5: *Pbest* is updated, if the current evaluation value is better than *Pbest*. *gbest* is determined as the best individual in the whole population. It is also updated as the best individual in the entire swarm.

Step 6: In the velocity update equation a stochastic Markov Jumping parameter is introduced. The particle jumps from its current state according the transition probability 0.9 for the initial current state and then the next state is predicted. A set of acceleration coefficients are pre-defined and a specific value is selected according to the current state automatically. Later on, getting all parameter values the velocity and position for all particles are then updated as the follows:

$$\begin{aligned} v_{ij}(t+1) = &\omega v_i(t) + c_1(\delta(t))rand_1(t)(pbest_{ij}(t) - P_{ij}(t)) \\ &+ c_2(\delta(t))rand_2(t)(gbest_{ij}(t) - P_{ij}(t)), \end{aligned} \tag{6.13}$$

$$P_{ij}(t+1) = P_{ij}(t) + v_{ij}(t+1) \tag{6.14}$$

$i = 1, 2, \cdots, n$, and $j = 1, 2, \cdots, m$, where n represent the number of particles and m represents the number of generating units in the system.

$P_{ij}(t+1)$ is the new updated position according to the new *Pbest* and *gbest* values. It is essential for newly updated *Pbestval* to be in the limits of its constraints. The population is initialized according to the constraints boundaries. Therefore, no risk of violation is involved in this case.

Step 7: *Pbest* and *gbest* are updated for all particles. The *gbest* obtained in the last iteration is the optimum solution for the problem.

Step 8: Repeat step 3 until the maximum number of iterations reach.

Step 9: go to Step 1: when T $\leq Hours_t$. Stop otherwise.

6.5 Experimental Setup and Simulation Results

The performance of proposed algorithms is examined for their applications in DELD non-smooth cost functions with also having non-linear constraints. The case-studies adopted in applications of the newly developed algorithm are 5-unit system with considering transmission losses having non-smooth cost function or valve-point loading effects, power balance constraints, ramp-rate limits and prohibited operating zones. The second case study used here is the 10 unit system with valve-point loading, power balance constraints and ramp-rate limits. The data used for simulating the above case-studies is taken from the published articles. All the experimental work is carried-out in Matlab R2015a, Intel Corei5 3.30GHZ 64bit Windows 10 operating system, with 8GB RAM. The proposed algorithms are compiled for 30 independent trials. The average/best results in terms of minimum cost values are tabulated and plotted. The parameters setting for the proposed algorithms and all other variants used in the experimental work are listed in the following Table 6.1. For comparison purposes we have re-implemented 10 widely used PSO variants, namely PSO with constriction factor (PSO-CF) [48], Comprehensive Learning (CLPSO) [4], Dynamic Multi-swarm PSO (DMS-PSO) [60], Combinatorial PSO (CPSO) [173], Genetic Algorithm (GA) [158], traditional PSO (PSO) [58], Unified Particle Swarm Optimization (UPSO) [174], Fully-informed PSO (FIPS) [59]. Fitness distance ratio based PSO algorithm (FDR-PSO), [175], PSO with constriction factor and local-best version (PSO-CF-Local) [56] and also the newly developed NS-SPSO. All the above mentioned PSO variants and GA algorithm were tested and evaluated by applying to the above mentioned two case studies of DELD problems. Due to the limitation of space we only plot the variants with the best performances in comparison to the proposed algorithms.

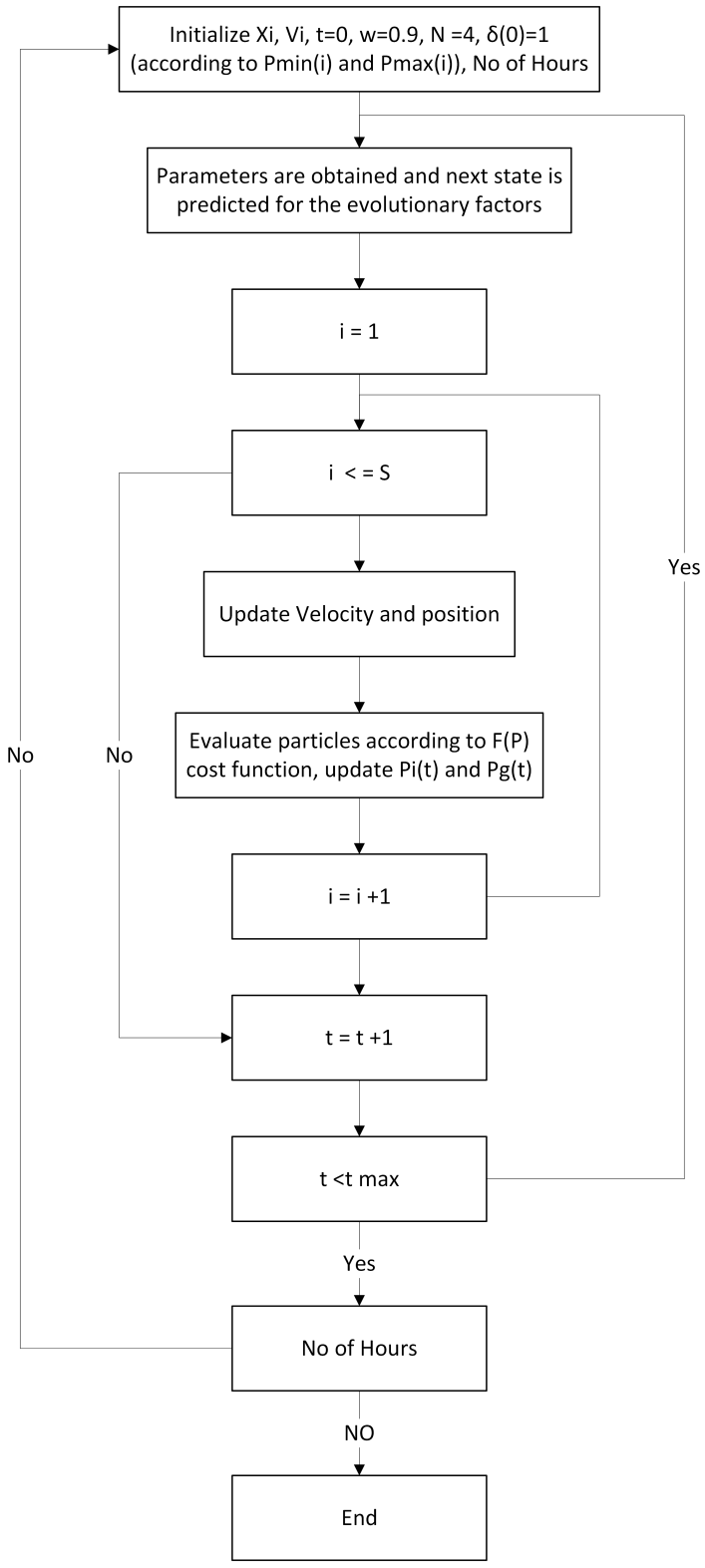


Figure 6.1: The Flowchart of NS-MJPSO Implementation to DELD problem

Table 6.1: Parameter settings of PSO for DELD applications

Algortihms	Acceleration	ω_{max}	ω_{min}	Cons; Factor (χ)	States
NS-MJPSO	C = [2, N]	0.9	0.5	$\phi = 0.9$	N = [4,8]
NS-SPSO	C = [2, N]	0.9	0.5		N = [4,8]
MPSO	$c_1, c_2 = 2$	0.9	0.4	1	
CPSO	$c_1, c_2 = 1.49$	0.9	0.7		
CL-PSO	$c_1, c_2 = 1.49445$				
FDR-PSO	$\phi_{1,2,3} = [1, 1, 2]$	0.9	0.5		

6.5.1 Sketch of Test Cases for simulations

6.5.1.1 Case 1: 5-Unit System

Table 6.2: Generating Units limits and cost coefficients

Unit	P_{min}	P_{max}	α(\$)	β(\$)	γ(\$)	e	f	U_r	D_r	$Pz_{min,1}$	$Pz_{max,1}$	$Pz_{min,2}$	$Pz_{max,2}$
1	10	75	0.008	2	25	100	0.042	30	30	10	10	10	10
2	20	125	0.003	1.8	60	140	0.04	30	30	20	20	20	20
3	30	175	0.0012	2.1	100	160	0.038	40	40	30	30	30	30
4	40	250	0.001	2	120	180	0.037	50	50	40	40	40	40
5	50	300	0.0015	1.8	40	200	0.035	50	50	50	50	50	50

Table 6.3: 24-Hours Load Demand (MW) for the 5-Units System

Hour	1	2	3	4	5	6	7	8	9	10	11	12
Load	410	435	475	530	558	608	626	654	690	704	720	740
Hour	13	14	15	16	17	18	19	20	21	22	23	24
Load	704	690	654	580	558	608	654	704	680	605	527	463

6.5.1.2 Case 2: 10-Unit System

6.5.2 Performance of Proposed NS-MJPSO and NS-SPSO

In this chapter, we present an application of the newly developed NS-MJPSO and NS-SPSO algorithms in DELD problem. As we have explained in detailed about the DELD that it is a non-smooth, non-linear, high dimension and complex multi-modal problem. To find the optimal solution of this problem is a challenging task. Basically, for comparison purposes we have re-implemented about 10 variants for the same problem. Each variant has its own advantages and disadvantages. Whereas, according to our analysis, the variants with having local search or neighbourhood topological structures and multi-swarms have the best performance for DELD. The DELD is a multi-modal problem having multiple

Table 6.4: NS-MJPSO solution for DELD 5 Unit System with transmission losses

Hour	P_{g1}	P_{g2}	P_{g3}	P_{g4}	P_{g5}	Total Gen	Load
1	43.01	60.61	127.71	97.32	84.78	413.44	410
2	47.46	54.04	101.00	107.63	128.75	438.87	435
3	19.88	76.61	115.52	94.70	173.02	479.72	475
4	46.80	58.93	84.87	143.36	202.00	535.96	530
5	43.46	75.30	88.63	134.59	222.67	564.66	558
6	48.12	88.34	126.63	165.34	187.26	615.70	608
7	40.38	84.77	155.17	134.88	218.97	634.16	626
8	36.90	81.58	119.25	179.90	245.45	663.09	654
9	33.21	86.99	125.37	205.23	249.35	700.15	690
10	37.97	86.01	146.65	190.91	252.90	714.44	704
11	38.29	80.21	131.39	207.03	274.14	731.06	720
12	42.52	70.23	156.25	231.58	250.96	751.55	740
13	43.29	89.74	160.66	189.68	230.99	714.35	704
14	60.60	85.59	129.75	162.33	261.80	700.07	690
15	65.08	71.37	124.70	166.14	235.70	662.98	654
16	53.02	94.54	112.08	139.81	187.60	587.04	580
17	40.83	75.91	108.98	154.03	184.76	564.51	558
18	34.67	90.37	139.81	151.46	199.39	615.70	608
19	45.80	87.67	118.69	183.99	226.89	663.04	654
20	35.82	92.86	152.84	233.54	199.40	714.47	704
21	45.79	84.27	135.37	209.49	214.81	689.73	680
22	32.11	89.41	149.08	170.69	171.32	612.61	605
23	40.68	74.04	115.38	155.33	147.33	532.76	527
24	37.24	62.91	138.34	109.08	119.81	467.38	463

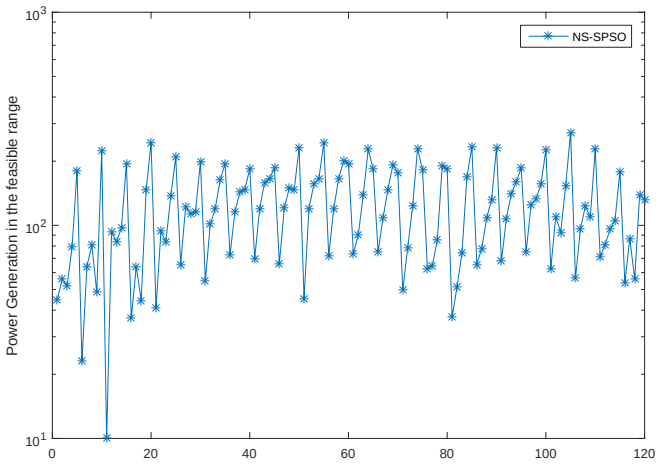


Figure 6.2: NS-SPSO convergence in the feasible boundaries for 24 hours dispatch

Table 6.5: NS-SPSO solution for DELD 5 Unit System with transmission losses

Hour	P_{g1}	P_{g2}	P_{g3}	P_{g4}	P_{g5}	Total Gen	Load
1	44.81	56.18	52.20	79.38	181.11	413.67	410
2	23.09	64.10	81.00	48.49	222.57	439.25	435
3	10.00	93.46	83.55	97.57	195.33	479.90	475
4	36.77	64.17	44.36	146.09	244.95	536.34	530
5	40.89	93.72	83.51	137.74	208.82	564.68	558
6	65.59	122.19	113.07	115.58	199.41	615.84	608
7	54.87	101.42	119.42	163.20	195.31	634.22	626
8	72.80	116.02	142.88	147.23	183.99	662.91	654
9	69.91	119.58	159.16	166.04	185.22	699.91	690
10	66.09	121.09	149.19	146.81	231.24	714.41	704
11	45.27	120.05	156.25	165.23	244.12	730.92	720
12	71.98	119.20	165.06	200.24	194.94	751.43	740
13	73.51	90.70	138.89	227.76	183.56	714.42	704
14	75.00	108.41	147.29	192.80	176.43	699.93	690
15	49.66	78.66	122.82	229.59	182.37	663.08	654
16	62.72	64.89	85.15	189.95	184.46	587.18	580
17	37.13	51.55	74.11	168.83	233.17	564.79	558
18	65.59	77.90	108.03	132.05	232.26	615.83	608
19	68.03	107.78	140.82	160.09	186.19	662.90	654
20	75.00	125.00	132.56	155.99	225.92	714.46	704
21	62.78	109.75	92.67	152.84	272.01	690.06	680
22	56.42	96.36	122.94	109.10	227.93	612.74	605
23	71.19	81.03	95.99	105.45	179.17	532.83	527
24	53.85	86.70	56.08	139.57	131.41	467.61	463

Table 6.6: Best Results for Case 1

Method	Minimum Cost ($ per 24 hr)	Comp; time	No. of Hits to the Global
NS-MJPSO	47312.3425	75	28
NS-SPSO	47833.9157	67	25
MPSO	47356.786	110	15
CPSO	46265.875	112	24
CL-PSO	48545.995	85	15
FDR-PSO	52377.896	105	23

Table 6.7: NS-MJPSO solution for DELD 10 Unit System

Load	P_{g1}	P_{g2}	P_{g3}	P_{g4}	P_{g5}	P_{g6}	P_{g7}	P_{g8}	P_{g9}	P_{g10}	Total Gen	Load
1	191.96	176.21	139.25	128.04	91.26	93.30	39.29	90.12	31.57	55.00	1036.00	1036
2	251.35	143.55	115.82	175.20	125.85	78.32	59.03	64.42	41.46	55.00	1110.00	1110
3	271.75	192.44	170.54	192.74	106.29	98.93	37.98	63.83	68.50	55.00	1258.00	1258
4	350.20	249.79	183.56	166.22	117.24	96.99	49.06	90.46	47.48	55.00	1406.00	1406
5	373.54	264.41	194.18	199.50	129.60	86.62	60.84	74.53	41.77	55.00	1480.00	1480
6	332.63	335.96	264.07	187.13	121.82	108.05	89.77	88.81	44.77	55.00	1628.00	1628
7	366.23	407.20	203.06	188.39	150.62	113.37	79.75	74.15	64.22	55.00	1702.00	1702
8	314.42	453.94	276.79	153.14	160.30	103.58	109.15	103.35	46.32	55.00	1776.00	1776
9	379.29	439.70	262.64	193.19	176.63	141.38	99.59	107.51	69.06	55.00	1924.00	1924
10	458.11	446.13	314.30	208.61	197.93	124.06	104.77	116.60	46.49	55.00	2072.00	2072
11	453.81	437.36	327.76	258.61	169.59	143.23	113.92	113.88	72.84	55.00	2146.00	2146
12	463.43	452.07	321.81	298.92	213.35	127.87	128.59	99.32	59.64	55.00	2220.00	2220
13	443.03	386.81	319.72	257.50	221.06	114.91	106.71	94.51	72.76	55.00	2072.00	2072
14	380.94	328.14	320.14	244.26	223.34	105.62	112.38	110.73	43.45	55.00	1924.00	1924
15	353.51	339.54	294.66	208.83	180.98	139.57	95.04	80.81	28.06	55.00	1776.00	1776
16	339.08	309.68	214.68	195.38	157.14	91.77	73.58	68.13	49.56	55.00	1554.00	1554
17	273.50	351.29	155.11	188.21	131.01	116.37	83.55	77.06	48.90	55.00	1480.00	1480
18	301.00	426.68	209.87	168.12	130.88	86.99	90.29	83.27	75.89	55.00	1628.00	1628
19	376.45	397.62	274.84	185.48	115.09	111.09	116.85	77.98	65.59	55.00	1776.00	1776
20	447.89	454.01	333.67	232.81	161.99	149.41	92.30	98.39	46.53	55.00	2072.00	2072
21	413.60	382.81	297.08	236.63	203.33	108.62	64.20	105.06	57.68	55.00	1924.00	1924
22	337.59	307.70	243.57	190.38	155.86	137.50	50.52	89.53	60.35	55.00	1628.00	1628
23	270.16	241.31	163.82	141.49	158.34	94.08	78.38	69.72	59.71	55.00	1332.00	1332
24	282.32	204.37	110.24	102.15	139.23	110.21	53.02	83.84	43.63	55.00	1184.00	1184

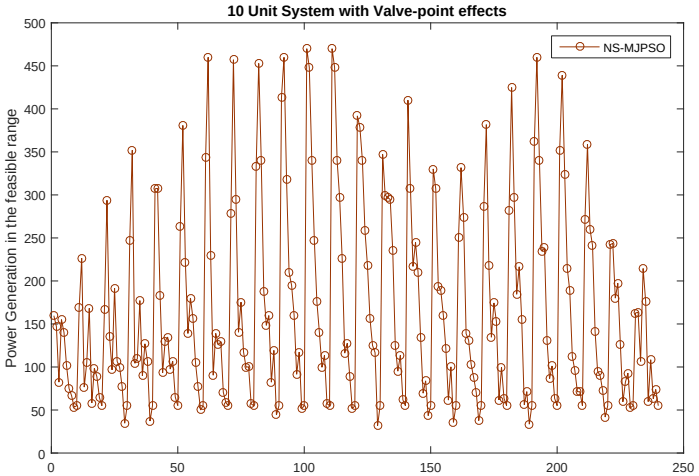


Figure 6.3: NS-MJPSO convergence in the feasible boundaries for 24 hours dispatch

Table 6.8: NS-SPSO solution for DELD 10 Unit System

Load	P_{g1}	P_{g2}	P_{g3}	P_{g4}	P_{g5}	P_{g6}	P_{g7}	P_{g8}	P_{g9}	P_{g10}	Total Gen	Load
1	159.62	147.03	82.39	155.11	140.16	101.74	74.98	66.71	53.25	55.00	1036.00	1036
2	168.62	226.52	76.65	105.14	168.30	58.12	98.35	88.83	64.48	55.00	1110.00	1110
3	167.28	293.87	135.72	96.58	191.57	106.71	99.64	76.97	34.66	55.00	1258.00	1258
4	247.24	352.01	104.32	109.64	177.24	89.60	126.96	106.80	37.19	55.00	1406.00	1406
5	307.07	307.70	182.82	93.11	130.18	134.82	97.60	106.64	65.08	55.00	1480.00	1480
6	263.25	380.82	221.37	138.94	180.18	156.62	104.85	76.96	50.02	55.00	1628.00	1628
7	343.15	460.00	230.14	90.43	139.07	125.63	129.33	70.60	58.65	55.00	1702.00	1702
8	278.70	457.12	294.74	140.41	175.41	117.34	99.78	100.26	57.24	55.00	1776.00	1776
9	333.69	452.86	340.00	188.29	148.40	159.84	82.36	119.34	44.23	55.00	1924.00	1924
10	413.55	460.00	318.22	209.93	194.87	160.00	91.53	117.40	51.49	55.00	2072.00	2072
11	470.00	448.08	340.00	246.70	175.71	139.83	99.66	113.61	57.41	55.00	2146.00	2146
12	470.00	448.36	340.00	296.59	225.71	116.11	127.57	89.15	51.51	55.00	2220.00	2220
13	391.88	378.76	339.90	258.79	217.91	156.15	125.07	116.74	31.82	55.00	2072.00	2072
14	347.12	299.76	296.86	294.73	235.04	125.01	95.13	113.55	61.80	55.00	1924.00	1924
15	410.16	307.37	216.86	244.74	210.17	134.71	69.57	83.92	43.49	55.00	1776.00	1776
16	330.23	307.04	194.15	188.79	160.23	121.18	61.51	100.19	35.68	55.00	1554.00	1554
17	250.40	331.75	273.97	138.81	131.14	102.97	87.64	70.66	37.67	55.00	1480.00	1480
18	286.89	381.74	218.09	134.60	174.47	152.55	61.63	99.55	63.50	55.00	1628.00	1628
19	281.94	424.83	297.17	184.51	216.49	154.97	56.15	71.41	33.55	55.00	1776.00	1776
20	361.94	460.00	340.00	234.51	238.42	131.15	86.15	101.23	63.55	55.00	2071.94	2072
21	351.75	439.12	323.83	215.01	188.42	112.28	96.10	71.25	71.24	55.00	1924.00	1924
22	271.79	359.14	259.99	241.10	141.63	95.00	90.09	72.48	41.79	55.00	1628.00	1628
23	242.25	243.66	180.04	197.33	126.28	59.53	82.58	92.81	52.51	55.00	1332.00	1332
24	162.78	163.64	106.50	214.54	176.27	59.75	108.47	63.15	73.90	55.00	1184.00	1184

Table 6.9: Best Results for Case 2

Method	Minimum cost ($ per 24 hr)	Comp; time	No. of Hits to the Global
NS-SPSO	1047553.394	60	27
NS-MJPSO	1036140.586	62	28
MPSO	1046725.908	85	18
CPSO	1132533.786	89	17
CL-PSO	1052866.956	69	14
FDR-PSO	1049055.566	72	20

local optimum. The second reason for selecting these variants is their best performance in constrained problems. DELD is highly constrained problem.

The emphasis in this study is the development of the best algorithm with good performance in the real-world environment. Due to the limitation of space we have presented the performance of the newly developed algorithm and the best results in terms of minimum cost of the existing methods have been taken for comparison. In Table 6.6 and Table 6.9 the comparative results of for Case Study 1 and Case Study 2 are given. The Minimum cost values illustrate the best performance of the participant methods. In both cases, the proposed NS-MJPSO has produced best results in terms of solution quality or accuracy and NS-SPSO has produced the best performance in terms of the shortest computation burden. The proposed methods have the ability of adaptation and robustness in complex, nonlinear real-world environment.

6.6 Summary of the Chapter

This chapter presents the basic knowledge about the Dynamic Economic Load Dispatch (DELD) problem. The analytical and heuristic methods that have already been used for the solution of DELD objective are summarised. The main objective of DELD problem is to minimize the total generation cost of the power system in the manner to satisfy the constraints applied to the system and generators. The problem is dynamic and changes the required load demand dynamically with time. The system has the line flow constraints and ramp-rate limits. The generator has prohibited operating zones and capacity limits constraints. These constraints make the problem more complex.

In Chapter 3 and Chapter 4, we have developed two new algorithms with the assumption of the best performance in real-world problems. In Chapter 5, we have examined the performance of both the algorithms in static ELD problem. While in this chapter, a more complex problem has been applied the proposed algorithms. The numerical results illustrate the average/best performance in terms of minimum cost value of the proposed algorithms. The proposed algorithms have the evolutionary adaptation strategy involved in the main process. We conclude this chapter with the contribution of a desirable solution to complex problem by using two new algorithms.

Chapter 7

Conclusions & Future Work

7.1 Conclusions

The present research is motivated by the wide use and satisfactory performance of the PSO algorithms that contribute substantially to the development of the evolutionary computation. As is well known, the PSO algorithms have their own advantages of light computation burden, simple implementation procedure and sub-optimal solutions for complex optimisation problems. In addition, the convergence of PSO algorithms has no dependency on the initialization. Owing to their computational merits, the PSO algorithms have been applied to a variety of practical problems.

Nevertheless, there are certain limitations with PSO algorithms that have been widely recognized. For example, a typical PSO algorithm has social, asocial (cognitive) and inertia weight parameters, and the control of the whole algorithm is totally dependent on these parameters. As such, the performance of the PSO algorithm is heavily dependent on the tuning of the parameters, which means that the overall performance could be extremely sensitive to the parameters which, in turn, will lead to certain limitations such as lack of diversity, premature convergence or stagnation of the particles in the local optimum as well as parameter adjustments. In this case, there appears to be an urgent need to look for some more robust, less sensitive yet more accurate modified PSO algorithms, and this is the main motivation of our research.

In this thesis, to improve the robustness as accuracy, we have proposed two novel algorithms, namely, N State Markov Jumping Particle Swarm Optimization (NS-MJPSO) and N State Switching Particle Swarm Optimization (NS-SPSO). Both the novel NS-MJPSO and NS-SPSO algorithms have, to some extent, addressed the previously identified limitations through theoretical analysis and numerical simulations. It has been shown that the present algorithms are indeed efficient, accurate and robust as compared to the traditional PSO algorithms.

Let us now summarise the advantages and applications of the proposed algorithms as follows.

- In the first proposed NS-MJPSO algorithm, a novel N-state Markov chain is introduced to the velocity update equation with hope to increase the accuracy as well as the robustness, where the parameter N plays an important role in dividing the state space based on the domain knowledge. Obviously, the bigger the N, the more accurate the results and the heavier the computation burden. Therefore, a proper choice of the parameter N is vitally critical.

  At each iteration, the particle is evaluated to determine the most adequate state it

is associated, and then a Markovian jumping probability is enforced to decide which is the next state according to the quantified evolutionary factor. After a series of empirical experiments, we assign the initial transition probability as 0.9 which gives us a good trade-off between the accuracy and the running time. During the state jumping or switching, according to the evaluated evolutionary factor at each step, the current state will jump to the next possible state with a given probability.

The performance of our proposed PSO algorithm is then examined by applying it to some benchmark functions. The results produced by NS-MJPSO are then compared with the existing state-of-the-art algorithms. For the benefits of the comparison, the average/best evaluation values of each algorithms are given both in tables and in graphical illustration for each benchmark function. It is shown that our proposed PSO algorithm has produced the best results in terms of the large improvement of the accuracy and at least the same level (sometimes much shorter) of the computation time.

- The proposed NS-MJPSO algorithm is shown to be successful based on the assumptions that 1) we know how to determine the jumping probability according to the a prior knowledge and 2) we don't really care about the computational burden induced by the extra stage of the Markovian state jumping. In practice, however, it is quite often that we have less domain knowledge about the optimization problem and the computational burden is a concern. In this case, we have proposed another novel PSO algorithm, which is the NS-SPSO algorithm. For NS-SPSO algorithm, we update the velocity purely based on the evolutionary factors where the state switches from one to another according to the evaluation of its evolutionary factor. In other words, the possibility for the state switching or staying is determined by how large the evolutionary factor is. Our proposed NS-SPSO algorithm is then examined through applications to some benchmark functions. The results produced by NS-SPSO are then compared with NS-MJPSO and other popular PSO algorithms. The average/best evaluation values of the algorithms are given in tables and also visualised in the graphical illustration for each function. The proposed algorithm has consumed the shortest computation time and the second best results in terms of the accuracy (NS-MJPSO is the best) in comparison to all other variants.

- To further demonstrate the application potential of our proposed PSO algorithms, in this thesis, we propose the power system operation problems as the application domain. The power system operation describes Unit Commitment (UC). In the UC problem, we further extract our objective to static and dynamic Economic Load

Dispatch (ELD) and (DELD) problems. These problems are complex, multi-modal and highly nonlinear with multi-local optimum. Several non-smooth constraints are applied with the objective functions. Both of our newly proposed algorithms are applied first to the simple static ELD problem whose aim is to find the solution for a single load. The proposed algorithms have produced promising results. Then, both our proposed algorithms are applied to the DELD problems, which are dynamic problems whose cost functions are dependent on time. The dispatch for 24 hours is selected. In a single trial the algorithm finds the optimal/economical solution for all the loads that optimizes all units satisfying constraints. The best performance is presented for both the algorithms in both the applications. It is concluded that the milestones of this research are achieved.

7.2 Limitations of this Thesis

The research of PSO development is classified into five categories by [14], where one category stands for the simplification of PSO algorithm. In other words, the basic aim with doing further improvement in PSO algorithm is to maintain it's simplicity. However, the main concern of this thesis is to work out the instability and inconsistency problem by producing a robust algorithm. We have achieved the objective of accuracy by introducing additional parameters to the structure of PSO algorithm. Subsequently, it causes the following limitation.

- Increasing complexity by involving additional parameters like number of states N, evolutionary factor, transition probabilities and population distribution. The accuracy increases by increasing the number of states and the complexity increases consequentially.

- As mentioned in the previous section, the performance of NS-MJPSO is based on the adequate assignment of jumping probability and number of states N. This required some prior knowledge of the domain to determine the values of parameters, which is not quite often the case in real-world problems.

- The newly developed are very flexible in terms of parameter changes because we have many values of each parameter. So, a small changes does not effect the performance of the algorithm. However, the problem is tuning and adjusting the parameter manually in the initialisation time. An automatic and adaptive mechanism is needed to make the process more easier and simpler.

7.3 Future Work

The future work for this thesis is classified into two sub-sections as follows:

7.3.1 Future Milestones for Algorithms

1. The first future milestone is to extend the work further starting from the existing stage. As we have proposed the N-state concept in both of the algorithms, there are certain parameters that have to be tuned manually, for example, the size N for both algorithms and the jumping probability for the NS-MJPSO algorithm, which would rely on the domain knowledge we have in hand. Obviously, there is a need to develop a mechanism to automatically or adaptively determine the size N and the jumping probability matrix that could be extracted from the problem-specific knowledge base for.

2. Some more efficient artificial learning mechanism is required to train the social and asocial (cognitive) parameters in order to understand the nature and dimensionality better. Again, an automated training scheme would be a great addition to make the developed new PSO algorithms even more efficient.

3. Some other state-of-the-art algorithms could be combined with our proposed ones in order to solve more complicated problems. For example, the deep learning algorithms could be applied first to learn the attributes/structure/classes of the problems addressed before they are converted to the optimization problems that can then be solved by the proposed PSO algorithms.

4. The next milestone for the future work is to deal with the many-objective optimization problems that usually refer to those problems with more than three competing objectives. The challenges would be 1) how to provide a new (decent) solution quickly; 2) how to make sure that the after-change solution is not very different from the before-change solution; 3) how to make sure that the new solutions are as close to a reference solution as possible; and 4) how to make sure that the future solutions must be in certain bounds? These are fundamental challenges that have not been adequately investigated in the computer science community yet.

7.3.2 Future Milestones for Applications

1. Power system is not the main emphasis of this thesis, but is purely used as a test bed. However, from the current studies we have some milestones for future research. For example, the current application is about single objective ELD and DELD problems. In the future, we intend to extend the work to multi-objective and many-objective problems, which can better reflect the environmental practice such as emission dispatch with combined ELD and emission objectives.

2. The second future target for applications is to apply the developed algorithms to other more sophisticated engineering problems such as the dynamic optimization problems for ship design process where the issues of ship routing and scheduling, inventory management, trajectory planning, vibration control and inner shell optimization have gained particular research interest.

Bibliography

[1] S.-Y. Ho, H.-S. Lin, W.-H. Liauh, and S.-J. Ho, "Opso: Orthogonal particle swarm optimization and its application to task assignment problems," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 38, no. 2, pp. 288–298, 2008. 4, 5, 35, 68, 69

[2] B. Liu, L. Wang, and Y.-H. Jin, "An effective pso-based memetic algorithm for flow shop scheduling," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 1, pp. 18–27, 2007. 4, 5, 35, 68, 69

[3] G. Ciuprina, D. Ioan, and I. Munteanu, "Use of intelligent-particle swarm optimization in electromagnetics," *Magnetics, IEEE Transactions on*, vol. 38, no. 2, pp. 1037–1040, 2002. 4, 5, 35, 69

[4] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *Evolutionary Computation, IEEE Transactions on*, vol. 10, no. 3, pp. 281–295, 2006. 4, 5, 18, 35, 36, 40, 47, 68, 69, 71, 77, 94, 123

[5] R. C. Eberhart and Y. Shi, "Guest editorial special issue on particle swarm optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 8, no. 3, pp. 201–203, 2004. 5, 16, 20, 35, 68

[6] Z.-H. Zhan, J. Xiao, J. Zhang, and W.-n. Chen, "Adaptive control of acceleration coefficients for particle swarm optimization based on clustering analysis," in *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pp. 3276–3282, IEEE, 2007. 5, 32, 36, 40, 69, 72

[7] Z.-H. Zhan, J. Zhang, Y. Li, and H.-H. Chung, "Adaptive particle swarm optimization," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 39, no. 6, pp. 1362–1381, 2009. 5, 6, 21, 29, 36, 40, 41, 45, 69, 70, 72

[8] Y. Tang, Z. Wang, and J.-a. Fang, "Parameters identification of unknown delayed genetic regulatory networks by a switching particle swarm optimization algorithm," *Expert Systems with Applications*, vol. 38, no. 3, pp. 2523–2535, 2011. 5, 6, 19, 21, 29, 36, 40, 41, 42, 45, 69, 70, 72, 74

[9] 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," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. 9, no. 2, pp. 321–329, 2012. 5, 22, 30, 36

[10] Z.-L. Gaing, "Particle swarm optimization to solving the economic dispatch considering the generator constraints," *Power Systems, IEEE Transactions on*, vol. 18, no. 3, pp. 1187–1195, 2003. 7, 24, 92, 93, 94, 95, 103, 107, 116

[11] J.-B. Park, Y.-W. Jeong, J.-R. Shin, and K. Y. Lee, "An improved particle swarm optimization for nonconvex economic dispatch problems," *Power Systems, IEEE Transactions on*, vol. 25, no. 1, pp. 156–166, 2010. 7, 26, 94

[12] V. K. Jadoun, N. Gupta, K. Niazi, and A. Swarnkar, "Nonconvex economic dispatch using particle swarm optimization with time varying operators," *Advances in Electrical Engineering*, vol. 2014, 2014. 7, 94

[13] B. Bahmani-Firouzi, E. Farjah, and R. Azizipanah-Abarghooee, "An efficient scenario-based and fuzzy self-adaptive learning particle swarm optimization approach for dynamic economic emission dispatch considering load and wind power uncertainties," *Energy*, vol. 50, pp. 232 – 244, 2013. 7, 28

[14] R. Cheng and Y. Jin, "A social learning particle swarm optimization algorithm for scalable optimization," *Information Sciences*, vol. 291, pp. 43–60, 2015. 7, 8, 16, 45, 47, 76, 94, 113, 134

[15] Z.-L. Gaing, "Constrained dynamic economic dispatch solution using particle swarm optimization," in *Power Engineering Society General Meeting, 2004. IEEE*, pp. 153–158, IEEE, 2004. 8, 25, 27, 94, 116, 118, 120

[16] A. Chowdhury, H. Zafar, B. Panigrahi, K. Krishnanand, A. Mohapatra, and Z. Cui, "Dynamic economic dispatch using lbest-pso with dynamically varying sub-swarms," *Memetic Computing*, vol. 6, no. 2, pp. 85–95, 2014. 8, 117

[17] C. Panigrahi, P. Chattopadhyay, R. Chakrabarti, and M. Basu, "Simulated annealing technique for dynamic economic dispatch," *Electric power components and systems*, vol. 34, no. 5, pp. 577–586, 2006. 8, 116, 120

[18] B. Panigrahi, V. R. Pandi, and S. Das, "Adaptive particle swarm optimization approach for static and dynamic economic load dispatch," *Energy conversion and management*, vol. 49, no. 6, pp. 1407–1415, 2008. 8, 27, 116, 120

[19] E. Polak, *Optimization: algorithms and consistent approximations*, vol. 124. Springer Science & Business Media, 2012. 11

[20] J. C. Spall, *Introduction to stochastic search and optimization: estimation, simulation, and control*, vol. 65. John Wiley & Sons, 2005. 11

[21] D. B. Fogel, *Evolutionary computation: toward a new philosophy of machine intelligence*, vol. 1. John Wiley & Sons, 2006. 13

[22] L. J. Fogel, A. J. Owens, and M. J. Walsh, "Artificial intelligence through simulated evolution," 1966. 13

[23] L. Shi and G. Xu, "Self-adaptive evolutionary programming and its application to multi-objective optimal operation of power systems," *Electric Power Systems Research*, vol. 57, no. 3, pp. 181–187, 2001. 13

[24] L. Lai and J. Ma, "Application of evolutionary programming to transient and subtransient parameter estimation," *Energy Conversion, IEEE Transactions on*, vol. 11, no. 3, pp. 523–530, 1996. 13

[25] I. Rechenberg, "Evolution strategy: Natures way of optimization," in *Optimization: Methods and applications, possibilities and limitations*, pp. 106–126, Springer, 1989. 13

[26] T. Hatanaka, K. Uosaki, H. Tanaka, and Y. Yamada, "System parameter estimation by evolutionary strategy," in *SICE'96. Proceedings of the 35th SICE Annual Conference. International Session Papers*, pp. 1045–1048, IEEE, 1996. 13

[27] J. Louchet, "Stereo analysis using individual evolution strategy," in *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, vol. 1, pp. 908–911, IEEE, 2000. 13

[28] G. W. Greenwood, A. K. Gupta, and K. McSweeney, "Scheduling tasks in multiprocessor systems using evolutionary strategies.," in *International Conference on Evolutionary Computation*, pp. 345–349, 1994. 13

[29] H. J. Bremermann, *The evolution of intelligence: The nervous system as a model of its environment*. University of Washington, Department of Mathematics, 1958. 14

[30] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992. 14

[31] L. Caponetti, N. Abbattista, and G. Carapella, "A genetic approach to edge detection," in *Image Processing, 1994. Proceedings. ICIP-94., IEEE International Conference*, vol. 1, pp. 318–322, IEEE, 1994. 14

[32] G. L. Bosco, "A genetic algorithm for image segmentation," in *Image Analysis and Processing, 2001. Proceedings. 11th International Conference on*, pp. 262–266, IEEE, 2001. 14

[33] T. N. Malik, A. ul Asar, M. F. Wyne, and S. Akhtar, "A new hybrid approach for the solution of nonconvex economic dispatch problem with valve-point effects," *Electric Power Systems Research*, vol. 80, no. 9, pp. 1128–1136, 2010. 14

[34] K. F. Man, K. S. TANG, and S. Kwong, *Genetic algorithms for control and signal processing*. Springer Science & Business Media, 2012. 14

[35] R. Lukac, B. Smolka, K. Martin, K. N. Plataniotis, and A. N. Venetsanopoulos, "Vector filtering for color imaging," *Signal Processing Magazine, IEEE*, vol. 22, no. 1, pp. 74–86, 2005. 14

[36] J. Kenndy and R. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, 1995. 3, 15, 20, 34, 37, 39, 68, 71, 116

[37] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the sixth international symposium on micro machine and human science*, vol. 1, pp. 39–43, New York, NY, 1995. 3, 15, 34, 37, 68, 71

[38] H. Fan and Y. Shi, "Study on vmax of particle swarm optimization," in *Proc. Workshop on Particle Swarm Optimization, Purdue School of Engineering and Technology*, Indianapolis, IN, Apr, 2001. 17

[39] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*, vol. 5, pp. 4104–4108, IEEE, 1997. 17

[40] M. Abido, "Optimal design of power-system stabilizers using particle swarm optimization," *Energy Conversion, IEEE Transactions on*, vol. 17, no. 3, pp. 406–413, 2002. 17, 24, 93

[41] M. Abido, "Optimal power flow using particle swarm optimization," *International Journal of Electrical Power & Energy Systems*, vol. 24, no. 7, pp. 563–571, 2002. 17, 24

[42] E. Ozcan and C. K. Mohan, "Particle swarm optimization: surfing the waves," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, vol. 3, IEEE, 1999. 17

[43] A. Ratnaweera, S. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *Evolutionary Computation, IEEE Transactions on*, vol. 8, no. 3, pp. 240–255, 2004. 17, 21, 35, 39

[44] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, pp. 69–73, IEEE, 1998. 17, 20, 37, 39, 70, 71

[45] Y. Shi and R. C. Eberhart, "Parameter selection in particle swarm optimization," in *Evolutionary programming VII*, pp. 591–600, Springer, 1998. 17, 35, 36, 37, 39

[46] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, vol. 3, IEEE, 1999. 17, 35, 37, 39, 47, 70, 74, 77

[47] M. Clerc, "The swarm and the queen: towards a deterministic and adaptive particle swarm optimization," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, vol. 3, IEEE, 1999. 18, 20

[48] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, vol. 1, pp. 84–88, IEEE, 2000. 18, 39, 123

[49] J. Robinson, S. Sinton, and Y. Rahmat-Samii, "Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna," in *Antennas and Propagation Society International Symposium, 2002. IEEE*, vol. 1, pp. 314–317 vol.1, 2002. 18, 69

[50] C.-F. Juang, "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 34, pp. 997–1006, April 2004. 18, 21

[51] F. Valdez, P. Melin, and O. Castillo, "Modular neural networks architecture optimization with a new nature inspired method using a fuzzy combination of particle swarm optimization and genetic algorithms," *Information Sciences*, vol. 270, pp. 143 – 153, 2014. 18, 69

[52] P. Shelokar, P. Siarry, V. Jayaraman, and B. Kulkarni, "Particle swarm and ant colony algorithms hybridized for improved continuous optimization," *Applied Mathematics and Computation*, vol. 188, no. 1, pp. 129 – 142, 2007. 18, 69

[53] W.-J. Zhang, X.-F. Xie, *et al.*, "Depso: hybrid particle swarm with differential evolution operator," in *IEEE International Conference on Systems Man and Cybernetics*, vol. 4, pp. 3816–3821, 2003. 18

[54] N. Higashi and H. Iba, "Particle swarm optimization with gaussian mutation," in *Swarm Intelligence Symposium, 2003. SIS '03. Proceedings of the 2003 IEEE*, pp. 72–79, April 2003. 18, 69

[55] B. Liu, L. Wang, Y.-H. Jin, F. Tang, and D.-X. Huang, "Improved particle swarm optimization combined with chaos," *Chaos, Solitons & Fractals*, vol. 25, no. 5, pp. 1261 – 1271, 2005. 18

[56] P. N. Suganthan, "Particle swarm optimiser with neighbourhood operator," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, vol. 3, IEEE, 1999. 18, 20, 39, 68, 123

[57] J. Kennedy, "Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, vol. 3, pp. –1938 Vol. 3, 1999. 18, 68

[58] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, vol. 2, pp. 1671–1676, 2002. 18, 47, 68, 77, 123

[59] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: simpler, maybe better," *Evolutionary Computation, IEEE Transactions on*, vol. 8, no. 3, pp. 204–210, 2004. 18, 39, 47, 68, 77, 123

[60] J. Liang and P. Suganthan, "Dynamic multi-swarm particle swarm optimizer," in *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE*, pp. 124–129, June 2005. 19, 47, 77, 123

[61] F. van den Bergh and A. Engelbrecht, "A cooperative approach to particle swarm optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 8, pp. 225–239, June 2004. 19

[62] R. Cheng, C. Sun, and Y. Jin, "A multi-swarm evolutionary framework based on a feedback mechanism," pp. 718–724, 2013. cited By 2. 19, 47, 77

[63] W. Du and B. Li, "Multi-strategy ensemble particle swarm optimization for dynamic optimization," *Information Sciences*, vol. 178, no. 15, pp. 3096 – 3109, 2008. Nature Inspired Problem-Solving. 19

[64] J. Kennedy, "Bare bones particle swarms," in *Swarm Intelligence Symposium, 2003. SIS '03. Proceedings of the 2003 IEEE*, pp. 80–87, April 2003. 19

[65] J. Zhou, "New look at creativity in the entrepreneurial process," *Strategic Entrepreneurship Journal*, vol. 2, no. 1, pp. 1–5, 2008. 19

[66] G. Iacca, F. Caraffini, F. Neri, and E. Mininno, "Single particle algorithms for continuous optimization," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pp. 1610–1617, June 2013. 19

[67] C. Sun, J. Zeng, J. Pan, S. Xue, and Y. Jin, "A new fitness estimation strategy for particle swarm optimization," *Information Sciences*, vol. 221, pp. 355 – 370, 2013. 19

[68] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 1, pp. 58–73, 2002. 20, 39

[69] I. C. Trelea, "The particle swarm optimization algorithm: convergence analysis and parameter selection," *Information processing letters*, vol. 85, no. 6, pp. 317–325, 2003. 20

[70] K. Yasuda, A. Ide, and N. Iwasaki, "Adaptive particle swarm optimization," in *Systems, Man and Cybernetics, 2003. IEEE International Conference on*, vol. 2, pp. 1554–1559, IEEE, 2003. 20

[71] V. Kadirkamanathan, K. Selvarajah, and P. J. Fleming, "Stability analysis of the particle dynamics in particle swarm optimizer," *Evolutionary Computation, IEEE Transactions on*, vol. 10, no. 3, pp. 245–255, 2006. 20

[72] F. Van den Bergh and A. P. Engelbrecht, "A study of particle swarm optimization particle trajectories," *Information sciences*, vol. 176, no. 8, pp. 937–971, 2006. 20

[73] R. C. Eberhart and Y. Shi, "Particle swarm optimization: developments, applications and resources," in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, vol. 1, pp. 81–86, IEEE, 2001. 20, 35, 37, 68, 70

[74] Y. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization," in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, vol. 1, pp. 101–106, IEEE, 2001. 20, 35

[75] P. Angeline, "Using selection to improve particle swarm optimization," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, 1998. 21, 40

[76] Y.-P. Chen, W.-C. Peng, and M.-C. Jian, "Particle swarm optimization with recombination and dynamic linkage discovery," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 6, pp. 1460–1470, 2007. 21, 40

[77] P. S. Andrews, "An investigation into mutation operators for particle swarm optimization," in *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pp. 1044–1051, IEEE, 2006. 21, 40

[78] Y. Lu, N. Zeng, Y. Liu, and N. Zhang, "A hybrid wavelet neural network and switching particle swarm optimization algorithm for face direction recognition," *Neurocomputing*, vol. 155, pp. 219 – 224, 2015. 22, 30

[79] N. Zeng, Y. Hung, Y. Li, and M. Du, "A novel switching local evolutionary pso for quantitative analysis of lateral flow immunoassay," *Expert Systems with Applications*, vol. 41, no. 4, Part 2, pp. 1708 – 1715, 2014. 22

[80] V. G. Gudise and G. K. Venayagamoorthy, "Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks," in *Swarm*

Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE, pp. 110–117, IEEE, 2003. 23

[81] S. Mohaghegi, Y. D. Valle, G. K. Venayagamoorthy, and R. G. Harley, "A comparison of pso and backpropagation for training rbf neural networks for identification of a power system with statcom," in *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE*, pp. 381–384, IEEE, 2005. 23

[82] J. Salerno, "Using the particle swarm optimization technique to train a recurrent neural model," in *ictai*, p. 0045, IEEE, 1997. 23

[83] Y. Song, Z. Chen, and Z. Yuan, "New chaotic pso-based neural network predictive control for nonlinear process," *Neural Networks, IEEE Transactions on*, vol. 18, no. 2, pp. 595–601, 2007. 23

[84] C.-M. Huang and F.-L. Wang, "An rbf network with ols and epso algorithms for real-time power dispatch," *Power Systems, IEEE Transactions on*, vol. 22, no. 1, pp. 96–104, 2007. 23

[85] U. Paquet and A. P. Engelbrecht, "Particle swarms for linearly constrained optimisation," *Fundamenta Informaticae*, vol. 76, no. 1-2, pp. 147–170, 2007. 23

[86] J. I. Ababneh and M. H. Bataineh, "Linear phase fir filter design using particle swarm optimization and genetic algorithms," *Digital Signal Processing*, vol. 18, no. 4, pp. 657–668, 2008. 23

[87] Y. Zhao and J. Zheng, "Particle swarm optimization algorithm in signal detection and blind extraction," in *Parallel Architectures, Algorithms and Networks, 2004. Proceedings. 7th International Symposium on*, pp. 37–41, IEEE, 2004. 23

[88] M. Tsatsanis and G. Giannakis, "Blind estimation of direct sequence spread spectrum signals in multipath," *Signal Processing, IEEE Transactions on*, vol. 45, pp. 1241–1252, May 1997. 23

[89] G. Tong, Z. Fang, and X. Xu, "A particle swarm optimized particle filter for nonlinear system state estimation," in *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pp. 438–442, 2006. 23

[90] L. Chih-Chin, "A novel image segmentation approach based on particle swarm optimization," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 89, no. 1, pp. 324–327, 2006. 23

[91] P. Ghamisi, M. Couceiro, F. Martins, and J. Atli Benediktsson, "Multilevel image segmentation based on fractional-order darwinian particle swarm optimization," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 52, pp. 2382–2394, May 2014. 23

[92] M. G. Omran, A. Salman, and A. P. Engelbrecht, "Dynamic clustering using particle swarm optimization with application in image segmentation," *Pattern Analysis and Applications*, vol. 8, no. 4, pp. 332–344, 2006. 23

[93] K. Chandramouli and E. Izquierdo, "Image classification using chaotic particle swarm optimization," in *2006 International Conference on Image Processing*, 2006. 23

[94] Y.-C. Chen, H.-C. Wang, and T.-J. Su, "Particle swarm optimization for image noise cancellation," in *Innovative Computing, Information and Control, 2006. ICICIC'06. First International Conference on*, vol. 1, pp. 587–590, IEEE, 2006. 23

[95] D. Van der Merwe and A. P. Engelbrecht, "Data clustering using particle swarm optimization," in *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, vol. 1, pp. 215–220, IEEE, 2003. 23

[96] M. Omran, A. P. Engelbrecht, and A. Salman, "Particle swarm optimization method for image clustering," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 19, no. 03, pp. 297–321, 2005. 23

[97] J. Varghese, M. Ghouse, S. Subash, M. Siddappa, M. S. Khan, and O. Bin Hussain, "Efficient adaptive fuzzy-based switching weighted average filter for the restoration of impulse corrupted digital images," *Image Processing, IET*, vol. 8, no. 4, pp. 199–206, 2014. 23

[98] H. Zhang and D.-y. LUO, "Status and development of study on blind image restoration algorithm [j]," *Journal of Image and Graphics*, vol. 10, p. 000, 2004. 23

[99] P.-Y. Yin, "Particle swarm optimization for point pattern matching," *Journal of Visual Communication and Image Representation*, vol. 17, no. 1, pp. 143–162, 2006. 23

[100] J.-G. Tang, X.-M. Zhang, Y.-L. Deng, Y.-X. Du, and Z.-Y. Chen, "Texture decomposition with particle swarm optimization method," *Computational materials science*, vol. 38, no. 2, pp. 395–399, 2006. 23

[101] A. Chatterjee, K. Pulasinghe, K. Watanabe, and K. Izumi, "A particle-swarm-optimized fuzzy-neural network for voice-controlled robot systems," *Industrial Electronics, IEEE Transactions on*, vol. 52, pp. 1478–1489, Dec 2005. 23

[102] Z. Bingül and O. Karahan, "A fuzzy logic controller tuned with pso for 2 dof robot trajectory control," *Expert Systems with Applications*, vol. 38, no. 1, pp. 1017–1031, 2011. 23

[103] V. Roberge, M. Tarbouchi, and G. Labonte, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time uav path planning," *Industrial Informatics, IEEE Transactions on*, vol. 9, pp. 132–141, Feb 2013. 23

[104] W. Jatmiko, K. Sekiyama, and T. Fukuda, "A pso-based mobile robot for odor source localization in dynamic advection-diffusion with obstacles environment: theory, simulation and measurement," *Computational Intelligence Magazine, IEEE*, vol. 2, pp. 37–51, May 2007. 23

[105] Y. del Valle, G. Venayagamoorthy, S. Mohagheghi, J.-C. Hernandez, and R. Harley, "Particle swarm optimization: Basic concepts, variants and applications in power systems," *Evolutionary Computation, IEEE Transactions on*, vol. 12, pp. 171–195, April 2008. 23

[106] M. AlRashidi and M. El-Hawary, "A survey of particle swarm optimization applications in electric power systems," *Evolutionary Computation, IEEE Transactions on*, vol. 13, pp. 913–918, Aug 2009. 23

[107] P. Onate Yumbla, J. Ramirez, and C. A. Coello Coello, "Optimal power flow subject to security constraints solved with a particle swarm optimizer," *Power Systems, IEEE Transactions on*, vol. 23, pp. 33–40, Feb 2008. 24

[108] E. Sortomme and M. El-Sharkawi, "Optimal power flow for a system of microgrids with controllable loads and battery storage," in *Power Systems Conference and Exposition, 2009. PSCE '09. IEEE/PES*, pp. 1–5, March 2009. 24

[109] H. Yoshida, K. Kawata, Y. Fukuyama, S. Takayama, and Y. Nakanishi, "A particle swarm optimization for reactive power and voltage control considering voltage security assessment," *Power Systems, IEEE Transactions on*, vol. 15, pp. 1232–1239, Nov 2000. 24, 93

[110] B. Zhao, C. Guo, and Y. Cao, "A multiagent-based particle swarm optimization approach for optimal reactive power dispatch," *Power Systems, IEEE Transactions on*, vol. 20, pp. 1070–1078, May 2005. 24

[111] Y. Fukuyama and H. Yoshida, "A particle swarm optimization for reactive power and voltage control in electric power systems," in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, vol. 1, pp. 87–93 vol. 1, 2001. 24

[112] Z. Bashir and M. El-Hawary, "Applying wavelets to short-term load forecasting using pso-based neural networks," *Power Systems, IEEE Transactions on*, vol. 24, pp. 20–27, Feb 2009. 24

[113] W.-C. Hong, "Chaotic particle swarm optimization algorithm in a support vector regression electric load forecasting model," *Energy Conversion and Management*, vol. 50, no. 1, pp. 105 – 117, 2009. 24

[114] Z.-L. Gaing, "A particle swarm optimization approach for optimum design of pid controller in avr system," *Energy Conversion, IEEE Transactions on*, vol. 19, no. 2, pp. 384–391, 2004. 24

[115] Y.-X. Jin, H.-Z. Cheng, J.-y. Yan, and L. Zhang, "New discrete method for particle swarm optimization and its application in transmission network expansion planning," *Electric Power Systems Research*, vol. 77, no. 3, pp. 227–233, 2007. 24

[116] H. Shayeghi, M. Mahdavi, and A. Bagheri, "Discrete pso algorithm based optimization of transmission lines loading in tnep problem," *Energy Conversion and Management*, vol. 51, no. 1, pp. 112–121, 2010. 24

[117] S. Kannan, S. M. R. Slochanal, P. Subbaraj, and N. P. Padhy, "Application of particle swarm optimization technique and its variants to generation expansion planning problem," *Electric Power Systems Research*, vol. 70, no. 3, pp. 203–210, 2004. 24

[118] D. P. Kothari, "Power system optimization," in *Computational Intelligence and Signal Processing (CISP), 2012 2nd National Conference on*, pp. 18–21, IEEE, 2012. 24

[119] A. A. Esmin, G. Lambert-Torres, and A. C. Z. De Souza, "A hybrid particle swarm optimization applied to loss power minimization," *Power Systems, IEEE Transactions on*, vol. 20, no. 2, pp. 859–866, 2005. 24

[120] V. Miranda and N. Fonseca, "Epso-evolutionary particle swarm optimization, a new algorithm with applications in power systems," in *Proc. of the Asia Pacific IEEE/PES Transmission and Distribution Conference and Exhibition*, vol. 2, pp. 745–750, Citeseer, 2002. 24

[121] J.-B. Park, K.-S. Lee, J.-R. Shin, and K.-S. Lee, "A particle swarm optimization for economic dispatch with nonsmooth cost functions," *Power Systems, IEEE Transactions on*, vol. 20, no. 1, pp. 34–42, 2005. 24, 25, 92, 94, 95, 96, 118, 119

[122] L. dos Santos Coelho and C.-S. Lee, "Solving economic load dispatch problems in power systems using chaotic and gaussian particle swarm optimization approaches," *International Journal of Electrical Power & Energy Systems*, vol. 30, no. 5, pp. 297–307, 2008. 24, 25, 93

[123] K. Swarup and S. Yamashiro, "Unit commitment solution methodology using genetic algorithm," *Power Systems, IEEE Transactions on*, vol. 17, no. 1, pp. 87–91, 2002. 25

[124] B. Zhao, C. Guo, and Y. Cao, "Dynamic economic dispatch in electricity market using particle swarm optimization algorithm," in *Intelligent Control and Automation, 2004. WCICA 2004. Fifth World Congress on*, vol. 6, pp. 5050–5054, IEEE, 2004. 25

[125] D. Jeyakumar, T. Jayabarathi, and T. Raghunathan, "Particle swarm optimization for various types of economic dispatch problems," *International Journal of Electrical Power & Energy Systems*, vol. 28, no. 1, pp. 36–42, 2006. 25

[126] L. dos Santos Coelho and R. A. Krohling, "Predictive controller tuning using modified particle swarm optimization based on cauchy and gaussian distributions," in *Soft computing: methodologies and applications*, pp. 287–298, Springer, 2005. 25

[127] K. Thanushkodi *et al.*, "A new particle swarm optimization solution to nonconvex economic dispatch problems," *Power Systems, IEEE Transactions on*, vol. 22, no. 1, pp. 42–51, 2007. 26

[128] A. I. Selvakumar and K. Thanushkodi, "Anti-predatory particle swarm optimization: solution to nonconvex economic dispatch problems," *Electric Power Systems Research*, vol. 78, no. 1, pp. 2–10, 2008. 26

[129] T. Niknam, "A new fuzzy adaptive hybrid particle swarm optimization algorithm for non-linear, non-smooth and non-convex economic dispatch problem," *Applied Energy*, vol. 87, no. 1, pp. 327 – 339, 2010. 26

[130] Y. Wang, J. Zhou, H. Qin, and Y. Lu, "Improved chaotic particle swarm optimization algorithm for dynamic economic dispatch problem with valve-point effects," *Energy Conversion and Management*, vol. 51, no. 12, pp. 2893–2900, 2010. 27

[131] X. Yuan, A. Su, Y. Yuan, H. Nie, and L. Wang, "An improved {PSO} for dynamic load dispatch of generators with valve-point effects," *Energy*, vol. 34, no. 1, pp. 67 – 74, 2009. 27

[132] A. Chowdhury, H. Zafar, B. Panigrahi, K. Krishnanand, A. Mohapatra, and Z. Cui, "Dynamic economic dispatch using lbest-pso with dynamically varying sub-swarms," *Memetic Computing*, vol. 6, no. 2, pp. 85–95, 2014. 28

[133] T. Niknam and F. Golestaneh, "Enhanced adaptive particle swarm optimisation algorithm for dynamic economic dispatch of units considering valve-point effects and ramp rates," *IET generation, transmission & distribution*, vol. 6, no. 5, pp. 424–435, 2012. 28

[134] B. Mohammadi-ivatloo, A. Rabiee, and M. Ehsan, "Time-varying acceleration coefficients {IPSO} for solving dynamic economic dispatch with non-smooth cost function," *Energy Conversion and Management*, vol. 56, pp. 175 – 183, 2012. 29

[135] A. Safari and H. Shayeghi, "Iteration particle swarm optimization procedure for economic load dispatch with generator constraints," *Expert Systems with Applications*, vol. 38, no. 5, pp. 6043–6048, 2011. 29

[136] N. Zeng, Y. Hung, Y. Li, and M. Du, "A novel switching local evolutionary {PSO} for quantitative analysis of lateral flow immunoassay," *Expert Systems with Applications*, vol. 41, no. 4, Part 2, pp. 1708 – 1715, 2014. 30

[137] H. Dawid, "A markov chain analysis of genetic algorithms with a state dependent fitness function," *Complex Systems*, vol. 8, no. 6, pp. 407–418, 1994. 31

[138] B. Fingleton, "Specification and testing of markov chain models: an application to convergence in the european union," *Oxford Bulletin of Economics and Statistics*, vol. 59, no. 3, pp. 385–403, 1997. 31

[139] X. Mao and C. Yuan, *Stochastic differential equations with Markovian switching*. World Scientific, 2006. 31

[140] Y. Tang, J.-a. Fang, M. Xia, and X. Gu, "Synchronization of takagi–sugeno fuzzy stochastic discrete-time complex networks with mixed time-varying delays," *Applied Mathematical Modelling*, vol. 34, no. 4, pp. 843–855, 2010. 31

[141] A. Mantawy, Y. Abdel-Magid, and S. Selim, "A simulated annealing algorithm for unit commitment," *Power Systems, IEEE Transactions on*, vol. 13, pp. 197–204, Feb 1998. 31

[142] D. Simopoulos, S. Kavatza, and C. Vournas, "Unit commitment by an enhanced simulated annealing algorithm," *Power Systems, IEEE Transactions on*, vol. 21, pp. 68–76, Feb 2006. 31

[143] J. Zhang, H.-H. Chung, and W.-L. Lo, "Clustering-based adaptive crossover and mutation probabilities for genetic algorithms," *Evolutionary Computation, IEEE Transactions on*, vol. 11, no. 3, pp. 326–335, 2007. 32, 40, 72

[144] M. A. Abido, "Environmental/economic power dispatch using multiobjective evolutionary algorithms," *Power Systems, IEEE Transactions on*, vol. 18, no. 4, pp. 1529–1537, 2003. 32

[145] N. Sinha, R. Chakrabarti, and P. Chattopadhyay, "Evolutionary programming techniques for economic load dispatch," *Evolutionary Computation, IEEE Transactions on*, vol. 7, no. 1, pp. 83–94, 2003. 32

[146] M. A. Abido, "Multiobjective evolutionary algorithms for electric power dispatch problem," *Evolutionary Computation, IEEE Transactions on*, vol. 10, no. 3, pp. 315–329, 2006. 32

[147] J. Kennedy, J. F. Kennedy, and R. C. Eberhart, *Swarm intelligence*. Morgan Kaufmann, 2001. 34, 68

[148] R. A. Krohling and L. dos Santos Coelho, "Coevolutionary particle swarm optimization using gaussian distribution for solving constrained optimization problems," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 36, no. 6, pp. 1407–1416, 2006. 35, 68

[149] Z. Wang, L. Hu, I. Rahman, and X. Liu, "A constrained optimization approach to dynamic state estimation for power systems including pmu measurements," in

Automation and Computing (ICAC), 2013 19th International Conference on, pp. 1–6, IEEE, 2013. 35, 68

[150] L. Hu, Z. Wang, I. Rahman, and X. Liu, "A constrained optimization approach to dynamic state estimation for power systems including pmu and missing measurements," *Control Systems Technology, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2015. 35, 68

[151] R. Cheng and M. Yao, "Particle swarm optimizer with time-varying parameters based on a novel operator," *Applied Mathematics & Information Sciences*, vol. 5, no. 2, pp. 33–38, 2011. 35

[152] J. Kennedy, "The particle swarm: social adaptation of knowledge," in *Evolutionary Computation, 1997., IEEE International Conference on*, pp. 303–308, IEEE, 1997. 39

[153] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-P. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization," *KanGAL report*, vol. 2005005, 2005. 40, 71

[154] B.-Y. Qu, P. Suganthan, and S. Das, "A distance-based locally informed particle swarm model for multimodal optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 17, no. 3, pp. 387–402, 2013. 69

[155] R. Brits, A. P. Engelbrecht, and F. van den Bergh, "Locating multiple optima using particle swarm optimization," *Applied Mathematics and Computation*, vol. 189, no. 2, pp. 1859–1883, 2007. 69

[156] T. O. Weber and W. A. Van Noije, "Design of analog integrated circuits using simulated annealing/quenching with crossovers and particle swarm optimization," *Simulated Annealing-Advances, Applications and Hybridizations*, 2012. 71

[157] C.-T. Su and G.-J. Chiou, "A fast-computation hopfield method to economic dispatch of power systems," *Power Systems, IEEE Transactions on*, vol. 12, no. 4, pp. 1759–1764, 1997. 93, 97

[158] A. Bakirtzis, V. Petridis, and S. Kazarlis, "Genetic algorithm solution to the economic dispatch problem," *IEE Proceedings-Generation, Transmission and Distribution*, vol. 141, no. 4, pp. 377–382, 1994. 93, 123

[159] K. P. Wong and Y. W. Wong, "Genetic and genetic/simulated-annealing approaches to economic dispatch," *IEE Proceedings-Generation, Transmission and Distribution*, vol. 141, no. 5, pp. 507–513, 1994. 93

[160] D. C. Walters and G. B. Sheble, "Genetic algorithm solution of economic dispatch with valve point loading," *Power Systems, IEEE Transactions on*, vol. 8, no. 3, pp. 1325–1332, 1993. 93, 97, 116

[161] P.-H. Chen and H.-C. Chang, "Large-scale economic dispatch by genetic algorithm," *Power Systems, IEEE Transactions on*, vol. 10, no. 4, pp. 1919–1926, 1995. 93, 97

[162] S. Naka, T. Genji, T. Yura, and Y. Fukuyama, "Practical distribution state estimation using hybrid particle swarm optimization," in *Power Engineering Society Winter Meeting, 2001. IEEE*, vol. 2, pp. 815–820 vol.2, 2001. 93

[163] R. K. Pancholi and K. Swarup, "Particle swarm optimization for security constrained economic dispatch," in *Intelligent Sensing and Information Processing, 2004. Proceedings of International Conference on*, pp. 7–12, IEEE, 2004. 93

[164] A. Selvakumar and K. Thanushkodi, "A new particle swarm optimization solution to nonconvex economic dispatch problems," *Power Systems, IEEE Transactions on*, vol. 22, pp. 42–51, Feb 2007. 93

[165] A. A. El-Ela, T. Fetouh, M. Bishr, and R. Saleh, "Power systems operation using particle swarm optimization technique," *Electric Power Systems Research*, vol. 78, no. 11, pp. 1906–1913, 2008. 94, 99

[166] P. Sriyanyong, "An approach to economic dispatch with multiple fuels based on particle swarm optimization," in *2011 INTERNATIONAL SYMPOSIUM ON COMPUTATIONAL MODELS FOR LIFE SCIENCES (CMLS-11)*, vol. 1371, pp. 327–336, AIP Publishing, 2011. 94

[167] V. K. Jadoun, N. Gupta, K. Niazi, A. Swarnkar, and R. Bansal, "Multi-area economic dispatch using improved particle swarm optimization," *Energy Procedia*, vol. 75, pp. 1087–1092, 2015. 94

[168] A. J. Wood and B. F. Wollenberg, *Power generation, operation, and control*. John Wiley & Sons, 2012. 97, 119

[169] W.-M. Lin, F.-S. Cheng, and M.-T. Tsay, "An improved tabu search for economic dispatch with multiple minima," *Power Systems, IEEE Transactions on*, vol. 17, no. 1, pp. 108–112, 2002. 116

[170] H.-T. Yang, P.-C. Yang, and C.-L. Huang, "Evolutionary programming based economic dispatch for units with non-smooth fuel cost functions," *Power Systems, IEEE Transactions on*, vol. 11, no. 1, pp. 112–118, 1996. 116

[171] V. R. Pandi and B. K. Panigrahi, "Dynamic economic load dispatch using hybrid swarm intelligence based harmony search algorithm," *Expert Systems with Applications*, vol. 38, no. 7, pp. 8509–8514, 2011. 117, 122

[172] A. Ghosh, A. Chowdhury, S. Sinha, A. V. Vasilakos, and S. Das, "A genetic lbest particle swarm optimizer with dynamically varying subswarm topology," in *Evolutionary Computation (CEC), 2012 IEEE Congress on*, pp. 1–7, IEEE, 2012. 117

[173] B. Jarboui, M. Cheikh, P. Siarry, and A. Rebai, "Combinatorial particle swarm optimization (cpso) for partitional clustering problem," *Applied Mathematics and Computation*, vol. 192, no. 2, pp. 337–345, 2007. 123

[174] K. E. Parsopoulos and M. N. Vrahatis, "Unified particle swarm optimization for solving constrained engineering optimization problems," in *Advances in natural computation*, pp. 582–591, Springer, 2005. 123

[175] T. Peram, K. Veeramachaneni, and C. K. Mohan, "Fitness-distance-ratio based particle swarm optimization," in *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*, pp. 174–181, IEEE, 2003. 123