DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

METAHEURISTIC OPTIMIZATION ALGORITHMS FOR SOLVING MULTIOBJECTIVE ECONOMIC DISPATCH PROBLEM

by
Mert Sinan TURGUT

June, 2015
İZMİR

METAHEURISTIC OPTIMIZATION ALGORITHMS FOR SOLVING MULTIOBJECTIVE ECONOMIC DISPATCH PROBLEM

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Master of Science
in Mechatronics Engineering Department, Mechatronics Engineering Program

by
Mert Sinan TURGUT

June, 2015
İZMİR

M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"METAHEURISTIC OPTIMIZATION ALGORITHMS FOR SOLVING MULTIOBJECTIVE ECONOMIC DISPATCH PROBLEM"** completed by **MERT SİNAN TURGUT** under supervision of **ASST. PROF. GÜLESER KALAYCI DEMİR** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Güleser KALAYCI DEMİR

Supervisor

Doç. Dr. Hasan ÖZTÜRK

(Jury Member)

Doç. Dr. Şeyda TOPALOĞLU

(Jury Member)

Prof.Dr. Ayşe OKUR

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENT

I would like to thank my thesis supervisor Asst. Prof. Güleser KALAYCI DEMİR for supporting me during every section of my thesis. Her guidance, patience and encoruagement helped me to finish this thesis in a short time.

I would also like to thank my family for their endless supports throughout my life.

Mert Sinan TURGUT

METAHEURISTIC OPTIMIZATION ALGORITHMS FOR SOLVING MULTIOBJECTIVE ECONOMIC DISPATCH PROBLEM

ABSTRACT

In this thesis, Artificial Cooperative Search (ACS) algorithm is incorporated with the Quadratic Approximation (QA) operator to solve the multiobjective Economic Emission Load Dispatch (EELD) problems with different generation units. ACS is a Swarm Intelligence (SI)-based metaheuristic algorithm, based on the interaction between prey and predator organisms in a habitat, is effective at global search; however, it does not perform so well at exploring promising regions. QA operator, on the other hand, is a non-derivative-based efficient local search method that finds the minimum of a quadratic hyperspace passing through three points in a D-dimensional space. Solving the EELD problems with the hybridized ACS-QA algorithm, as being proposed in the present thesis, leads to more accurate results with fewer function evaluations. Also, multiobjectivity of the problem is handled by transforming it into a single-objective problem by using the Weighted Sum Method (WSM). The efficiency of the proposed ACS-QA algorithm is tested in comparison to the algorithms existing in literature by implementing it on six different benchmark optimization problems. Afterwards, the proposed ACS-QA algorithm and the ACS algorithm are implemented on multiobjective EELD problems with different generation units. The results are compared with the solutions in literature utilizing different metaheuristic optimization algorithms. Both studies firmly showed that the ACS-QA algorithm is able to find more accurate results even though it uses fewer function evaluation calls.

Keywords: Metaheuristics, Multiobjective optimization problems, Economic Emission Load Dispatch problem.

ÇOKLU AMAÇLI EKONOMİK EMİSYON YÜK DAĞITIMI PROBLEMİNİN ÜSTSEZGİSEL OPTİMİZASYON ALGORİTMALARI İLE ÇÖZÜMÜ

ÖZ

Bu tezde, Artificial Cooperative Search (ACS) algoritması ile Quadratic Approximation (QA) operatörü, değişik üretim ünitelerinden oluşan çoklu amaçlı Ekonomik Emisyon Yük Dağıtımı (EEYD) probleminin çözümü için birleştirilmiştir. Bir habitattaki av-avcı ilişkisine dayalı, Sürü Zekası (SZ) tabanlı üstsezgisel bir optimizasyon algoritması olan ACS, global arama konusunda etkilidir ama önemli bölgeleri aramada iyi performans gösterememektedir. Bir diğer yandan, D-boyutlu bir uzaydaki üç noktanın üzerinden geçen bir kuadratik süperuzayın minimumunu bulan ve türev tabanlı olmayan QA operatörü efektif bir yerel arama algoritmasıdır. Bu tezde de önerildiği gibi, çoklu amaçlı EEYD problemlerini hibritleştirilmiş ACS-QA algoritması ile çözmek, daha kesin sonuçlara daha az fonksiyon değerlendirilmesi ile ulaşılmasını sağlar. Ayrıca, problemdeki çoklu amaçlılık, Ağırlıklı Toplam Metodu (ATM) ile problemi tek amaçlı hale dönüştürerek ele alınmıştır. ACS-QA algoritmasının etkinliği, ACS-QA ile literatürdeki diğer farklı algoritmaları, altı farklı kıyaslamalı optimizasyon problemlerine uygulanıp karşılaştırılarak test edilmiştir. Sonra, önerilen ACS-QA ve ACS algoritmaları değişik üretim ünitelerinden oluşan çoklu amaçlı EEYD problemine uygulanmıştır. Sonuçlar, literatürde yer alan farklı üstsezgisel optimizasyon algoritmalarının uygulanıp elde edilen sonuçlar ile karşılaştırılmıştır. İki çalışma da göstermiştir ki, ACS-QA algoritması daha kesin sonuçları daha az fonksiyon değerlendirmesi yapmasına rağmen bulmuştur.

Anahtar kelimeler: Üstsezgisel algoritmalar, Çoklu amaçlı optimizasyon problemleri, Ekonomik Emisyon Yük Dağıtım problemi.

CONTENTS

Page

THESIS EXAMINATION RESULT FORM ii
ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZ v
LIST OF FIGURES viii
LIST OF TABLES x

CHAPTER ONE – INTRODUCTION 1

1.1 Overview of the Problem 1
1.2 Literature Review 2
1.3 Thesis Objectives and Contrubitions 5
1.4 Organization of the Thesis 6

CHAPTER TWO – PROBLEM FORMULATION 7

2.1 Economic Load Dispatch 7
2.2 Load Dispatching 7
2.3 Generation Scheduling 8
2.4 Generator Cost Functions 8
2.5 Emission Dispatch 11
2.6 Constraints 12
2.7 System Variables and Problem of Optimum Dispatch 13
2.8 The Multiobjective Economic Emission Load Dispatch Problem 14
2.8.1 Multiobjective Programming 14
2.8.1.1 Scalarizing Multiobjective Optimization Problems 16
2.8.1.2 No-Preference Methods 17
2.8.1.3 A Priori Methods 18
2.8.1.4 A Posteriori Methods 18
2.8.1.5 Interactive Methods 19

2.8.2 Solution of the Multiobjective EELD problem with WSM Method 19

CHAPTER THREE – OPTIMIZATION ALGORITHMS 22

3.1 Optimization 22
3.2 Optimization Algorithms 24
3.3 Metaheuristics 26
3.3.1 History of Metaheuristic Algorithms 27
3.3.2 Particle Swarm Optimization 28
3.4 Types of Optimization Problems 32
3.5 Constraint Handling and Penalty Method 37
3.6 Artificial Cooperative Search Algorithm 38
3.7 Quadratic Approximation Operator 42

CHAPTER FOUR – THE PROPOSED ACS-QA ALGORITHM 44

CHAPTER FIVE – NUMERICAL ANALYSIS AND SIMULATION RESULTS 50

5.1 Case Study 1 50
5.2 Case Study 2 53
5.3 Case Study 3 58
5.4 Case Study 4 62
5.5 Case Study 5 63

CHAPTER SIX – CONCLUSION 68

REFERENCES 70

APPENDICES 77

LIST OF FIGURES

Page

Figure 2.1 Simple model of a fossil plant. 9
Figure 2.2 Operating cost of a fossil generator 9
Figure 2.3 Operating cost curve with valve-point effects 11
Figure 2.4 Results of an example pareto optimization study, where the conflict between f_1 and f_2 is explored. In this example, rank 1 is a nondominated set while rank 2 and rank 3 are higher determined. The true pareto front is drawn with solid blue line 14
Figure 2.5 Graphical illustration of linear scalarization 17
Figure 3.1 Classification of optimization algorithms 24
Figure 3.2 A basic particle swarm optimization algorithm 30
Figure 3.3 Classification of optimization problems 32
Figure 3.4 A simple multiobjective optimization problem where two objectives conflict 33
Figure 3.5 The plot of $g(x)$ as the function of $f(x)$ as x varies from -3.4 to 0.8. The solid line is the Pareto front 34
Figure 3.6 A simple unconstrained minimization problem 35
Figure 3.7 A simple constrained optimization problem 36
Figure 3.8 Pseudo code of the ACS algorithm 41
Figure 4.1 Flowchart of the proposed algorithm 49
Figure 5.1 Convergence rates of the original ACS and the proposed ACS-QA algorithms for the Griewank function 52
Figure 5.2 Convergence rates of the original ACS and the proposed ACS-QA algorithms for the Ackley function 52
Figure 5.3 Convergence rates of the original ACS and the proposed ACS-QA algorithms for the Sphere function 53
Figure 5.4 Single Line Diagram of IEEE-30 bus 6 generators system 55
Figure 5.5 Pareto curves of the original ACS and proposed ACS-QA algorithms for the 6 generation units system 58
Figure 5.6 Pareto curves of the original ACS and proposed ACS-QA algorithms for the 10 generation units system 62

Figure 5.7 Pareto curves of the original ACS and proposed ACS-QA algorithms for the 11 generation units system 63
Figure 5.8 Pareto curves of the original ACS and proposed ACS-QA algorithms for the 40 generation units system 67

LIST OF TABLES

Page

Table 5.1 Statistical properties of the solutions of unconstrained benchmark optimization problems 51

Table 5.2 Best Compromised Solutions of 6-unit system 56

Table 5.3 Solutions of single-objective 6-units Economic Dispatch and Emission Dispatch problems 57

Table 5.4 Best compromised solutions of 10-unit system 60

Table 5.5 Solutions of single-objective 10-units Economic Dispatch and Emission Dispatch problems 61

Table 5.6 Best compromised solutions of 11-units system 63

Table 5.7 Best compromised solutions of 40-units system 65

Table 5.8 Solutions of single-objective 40-units Economic Dispatch and Emission Dispatch problems 66

Table 5.9 Solutions of single-objective 40-units Economic Dispatch and Emission Dispatch problems 67

CHAPTER ONE
INTRODUCTION

1.1 Overview of the Problem

Since an engineer always deals with the cost of products and services, the efficient optimum electric generation and and planning of electrical power generation have occupied an important position in electric power industry. Increasing power generation costs, shortage of energy resources and hazardous by-products of energy generation necessitate optimal economic dispatch and the introduction of renewable energy in modern power systems. A small percentage of saving in the operation of the system may correspond huge amount of reduction in operating cost and quantities of fuel consumed. Therefore, main objective of the economic dispatch problems is to minimize the operating costs of generating systems.

Electrical power system operation should be defined by reliability, security and economy (Rajkumar, 2014). Economic load dispatch (ELD), which minimizes the operating cost of the generation system, is considered one of the most important electrical power system operational planning problems.

Economic load dispatch (ELD) problem deals with meeting the load demand by optimizing the operation of power generation units at the minimum operating cost while satisfying system equality and inequality constraints (Aydin, Ozyon, Yasar & Liao, 2014). It is considered as the most reliable, efficient and low-cost power system dispatching problem (Bhattacharjee, Bhattacharya & Halder nee Dey, 2014). ELD problem is multimodal, highly non-linear and discontinuous. Factors such as valve-point effects, prohibited operating zones, ramp rate limits makes the problem highly non-linear although cost curve of generating units is are generally modelled as a smooth curve. Large steam turbine generators usually have multiple valves to control the active power balance. However, this opening and closing causes ripples in the cost function. This is known as valve-point loading effect. Considering the valve-point effects in ELD problems is important because ignoring them may lead to inaccuracy in generation dispatch. Other than that, generation units may have to work in certain ranges due to its physical capabilities such as machine components, vibration in shaft bearings, etc. Such limits are known as Prohibited Operating Zone (POZ). Furthermore, operating range of the generation units is limited by their ramp rate limits.

Recently, various governments have constituted environmental pollution acts that limit the emission of hazardous gases coming from power plants that burn fossil fuels. Thus emerged the economic emission dispatch (EED) problem in recent decades due to its superiority in minimizing the emission of harmful pollutants such as NO_x, SO_x, CO_x. Furthermore, since consideration of only minimum generation cost or minimum emission level may not be a coveted measure, economic emission load dispatch (EELD) comes up for the aim at minimizing both energy generation cost and pollutant gas emission level. EELD problem is a combination of economic load dispatch and economic emission dispatch problems. EELD is a multi-objective problem in nature, and it does not have an exact solution unless weights of the objectives or exact preferences are known (Rajesh, Abhinav, Rudesh & Panda, 2012). Therefore, it is needed to find compromising solutions, also known as pareto optimal solutions, which indicate the trade-off between two contradictory objectives (Shayeghi, 2014; Soumitra, 2012).

1.2 Literature Review

Economic dispatch (ED) problems are highly non-linear, correlated and multi-dimensional problems since valve-point effects and other constraints are present. Thus, conventional derivative-based optimization algorithms are not very effective in obtaining global optimum solution (Roy & Bhui, 2013) though variety of conventional derivative-based methods such as lambda iteration, lagrangian relaxation (Wood, Woolenberg & Sheble, 2013), integer programming (Dillon, Edwin, Kochs & Taud, 1978) and quadratic programming (Papageorgiou & Fraga, 2007) have been proposed in the literature. To this end, many researchers have proposed heuristic optimization techniques to deal with the disadvantages of the conventional methods.

Bhattacharjee et al. (2014) used Real Coded Chemical Reaction algorithm (RCCRO), which mimics the chemical interaction of molecules in a reaction to solve multi-objective EELD problem. During a chemical reaction, the molecules of initial reactants stay in unstable and high-energy states and exposed to a series of collusions with the other molecules and walls of the container. The reactants pass through energy barriers and eventually reach to low-energy stable states, which lead to become the final products. Chemical Reaction Optimization (CRO) is based on this occurance of driving the unstable high-energy states to low-energy stable states. CRO is designed to be a discrete optimization algorithm and has been proved to be a successful one. Recently, (Lam et al., 2012) have developed a real-coded version of the CRO algorithm, which is suitable for continuous optimization problems. In this paper, RCCRO algorithm is utilized as a non-dominated sorting algorithm. At each run, an optimal set of solutions is obtained and the most appropriate optimal solution is selected as the non-dominated set. Results show that efficiency and effectiveness of the algorithm is better than most of the algorithms in the literature however better efficiency and effectiveness measures can be achieved with different optimization algorithms and multiobjectivity handling methods.

Jeddi and Vahidinasab (2014) used a new modified harmony search algorithm (MHSA) based on a new memory consideration scheme that relies on the roulette wheel mechanism and a new improvising method based on wavelet mutation. Harmony Search Algorithm (HSA) is a new metaheuristic algorithm based on searching for the perfect state of harmony and developed by Lee and Geem (2005). HSA algorithm has also applied to many different optimization problems and proved to be a successful algorithm. Proposed MHSA algorithm in the paper improves the accuracy, robustness and convergence speed of the traditional HSA algorithm. In the MHSA algorithm, a new selection scheme for memory consideration is proposed and the process of generating random variable is adopted from the idea of mutation operator in the GA. The proposed algorithm is applied to both ELD and multiobjective EELD problems considering all practical constraints. Although favourable and better solutions are obtained from the other algorithms used in the literature, no information is given about the efficiency of the algorithm and better results can be achieved by utilizing different algorithms.

Aydin et al. (2014) utilized a new artificial bee colony algorithm with dynamic population size (ABCDP) to solve EELD problem with different IEEE bus systems. Roy and Bhui (2013) applied a non-dominated sorting quasi-oppositional teaching learning based algorithm (QOTLBO) to multi-objective EELD problem. Shayeghi and Ghasemi (2014) applied a modified artificial bee colony algorithm (ABC) which uses chaotic local search (CLS) to improve the searching ability with a new method based on fuzzy set theory, which is applied to select the most desirable solution from a Pareto-set to multi-objective EELD problem with three conflicting objectives. Rajasomashekar and Aravindhababu (2012) developed a biogeography based optimization algorithm to obtain best-compromised solution of multi-objective EELD problem. Hota and Barisal (2010) used a fuzzy-based bacterial foraging algorithm to extract the most desirable solution from the trade-off curve of the EELD problem. Zhang, Gong and Ding (2012) applied a bare-bones multi-objective particle swarm optimization algorithm (BB-MOPSO), which also uses external repository of elite particles and crowding distance techniques to extract the best compromise solution from a pareto-curve. Roy and Hazra (2014) used chemical reaction optimization (CRO) to solve EED problem for wind-fossil-fuel-based power system. Jadoun, Gupta, Niazi and Swarnkar (2014) applied dynamically controlled particle swarm optimization to large-scale nonconvex ED problems. Rajagopalan, Sengoden and Govindasamy (2014) utilized self-adaptive differential harmony search algorithm to solve ELD problems.

Also applied to ED problems are metaheuristic algorithms such as tabu search algorithm (Khamsawang & Jiriwibhakorn, 2009), particle swarm algoritm (Gaing, 2003), scatter search algorithm (Silva, Klen, Mariani & Coelho, 2013), seeker optimization algorithm (Sivasubramani & Swarup, 2010) and charged system search algorithm (Ozyon, Temurtas, Durmus & Kuvat, 2012).

In the literature, multi-objective EELD problem has been solved either directly or converting it into a single objective problem. Examples are improved PSO (Abido, 2009), NSGA (Abido, 2003a), NPGA (Abido, 2003b), SPEA (Abido, 2003c) and fuzzy-based bacterial foraging algorithms (MBFA) (Hota et al., 2010).

1.3 Thesis Objectives and Contrubitions

The objectives of this thesis are as follows:

- To solve the multiobjective EELD problems by considering valve-point effects and constraints such as power balance, power loss, generator capacity limits and prohibited operating zones by utilizing hybridized Artificial Cooperative Search-Quadratic Approximation algorithm.
- To present a hybridized ACS-QA algorithm that improves the classical ACS in accuracy, convergence speed and robustness.
- To apply different metaheuristics algorithms in the literature to some benchmark unconstrained optimization problems and test the efficiency of the algorithms by comparing the results.
- To analyze the pareto-optimal front generated by utilizing the optimization algorithms based weighted sum method to the problem.

In the present thesis, Artificial Cooperative Search algorithm (ACS) (Civicioglu, 2013), based on the interaction between predator and prey organisms in a habitat, is incorporated with Quadratic Approximation operator (QA) (Deep & Bansal, 2009), a non-derivative-based local search algorithm, to solve the multi-objective EELD problem for different IEEE bus systems. ACS is an effective global search algorithm for solving highly non-linear optimization problems; however, it performs not so well at searching promising regions. QA, on the other hand, is a local search method that finds the minima of a quadratic hyperspace. The approach in this thesis is to hybridize ACS and QA such that the proposed algorithm (to be called ACS-QA hereon) makes use of the global search capabilities of the ACS and the local search intensification properties of QA. By doing this hybridization, we provide an effective (requiring fewer function evaluations) and an efficient (obtaining more accurate results) algorithm for solving the EELD problems. The multi-objective nature of the EELD problem is handled by transforming it into single-objective problem by using Weighted Sum Method (WSM) (Rajasomashekar & Aravindhababu, 2012).

1.4 Organization of the Thesis

The thesis is organized as follows. The formulations of the ELD, EED and EELD problems are described in Chapter 2. The algorithms ACS and QA are discussed in detail in Chapter 3. Chapter 4 is devoted to construction and implementation of the proposed algorithm. Numerical examples and simulation results are given in Chapter 5. Finally, Chapter 6 concludes the thesis.

CHAPTER TWO
PROBLEM FORMULATION

2.1 Economic Load Dispatch

The ELD problem is based on allocating electricity production to generation units most effectively and economically to meet the electricity demand (Kaur, 2011). For an interconnected system, minimizing the cost is necessary. The ELD problem is utilized to define the production level of each generation unit to keep to cost of generation and transmission minimum for a prescribed schdule of load. The main objective of the problem is minimizing the cost of generation.

Multi-objective EELD problem considered in this thesis deals with minimization of both fuel cost and environmental emission while satisfying equality and inequality constraints of the generator units. Cost and emission objectives are independent from each other, making the problem bi-objective.

Multiple generators work to satisfy the consumer demand in a typical power system. Each generation unit usually has a unique cost-per-hour characteristic for its output operating range. For example, a station can has incremental operating costs for fuel and maintenance and fixed costs associated with the station is quite substantial for a nuclear power plant. Things can even get more complex for utilities if we count in transmission line losses and seasonal changes associated with hydroelectric plants.

2.2 Load Dispatching

The operation of a modern power generation system has become very complicated. It is necessary to maintain in the frequency and voltage limits while ensuring the reliability power supply. Also, matching the generation of active and reactive power with the load demand is another problem. For ensuring the reliability of a power system, it is necessary to implement some extra generation units to the system in case of outage of generation equipment. Furthermore, it is also beneficial to keep the electricity generation costs at minimum. The load dispatch center controls the interconnected network as a whole. The load dispatch center allocates the electricity power generation to each grid depending on the demand on that area. Each load dispatch centre controls load and frequency of its own by matching generation in different generating stations with total required electricity power demand plus power losses.

2.3 Generation Scheduling

In a power system, the power plants are all different distances away from the centre of loads and fuel costs are all different. Also under normal operating condition, generation capacity is more than the total demand plus losses. Therefore, there can be many different options for scheduling generation. In an interconnected power system, the main objective is to find the power scheduling of the each generation unit in such a way that operation cost is minimum.

The objective function, also known as cost function, may present economic fuel cost system or other objectives such as emission. The transmission loss formula can be derived and the loss coefficient are known as B-coefficients.

The ELD problem assumes that the amount of electric power to be supplied by a set of units is constant for a given interval of time and tries to minimize the cost of supplying this energy subject to constraints. Thus, it deals with the minimization of total cost of the system and constraints over the entire dispatch time.

2.4 Generator Cost Functions

The total operating cost of a generation unit includes the fuel cost, cost of labour and mainentance. Generally, cost of labour, supplies and maintenance are fixed percentages of incoming fuel costs. The power output of fossil plants increases greatly if a set of valves to its steam turbine at the inlet is opened. Figure 2.1 shows the simple model of a fossil plant.

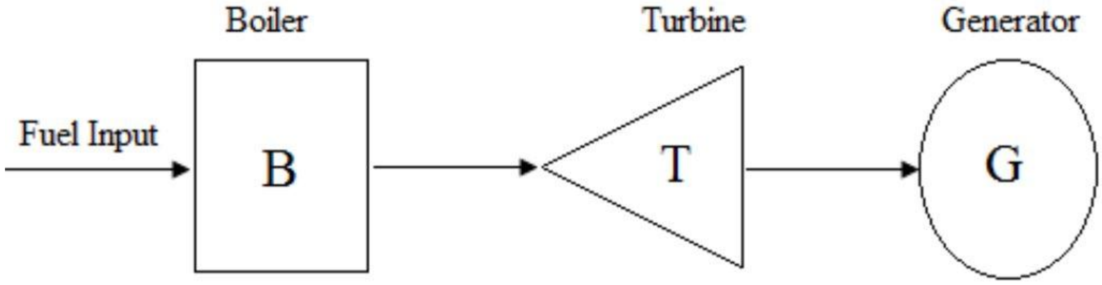


Figure 2.1 Simple model of a fossil plant (Kaur, 2011)

The operating cost of a plant is shown Figure 2.2. For practical purposes, the cost is generally approximated by one or two quadratic segments. Thus, it is practical to assume the cost in quadratic form.

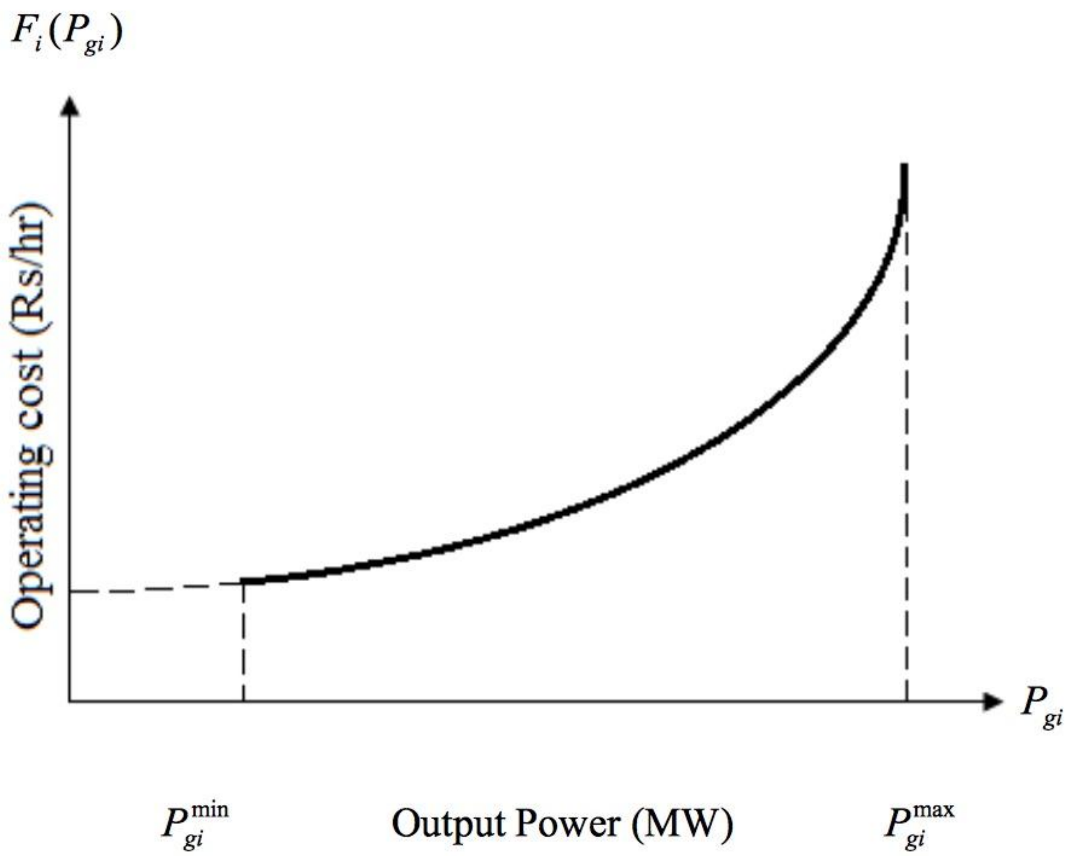


Figure 2.2 Operating cost of a fossil generator (Kaur, 2011)

Therefore, total cost of the system can be represented as follows,

$$C_F(P_G) = \sum_{i=1}^{N} (a_i + b_i P_{Gi} + c_i P_{Gi}^2) \tag{2.1}$$

where $C_F(P_G)$ is the total fuel cost in \$/h, a_i, b_i and c_i are the cost coefficients for the *i*-th generator, values of those coefficients are given in Appendix B. N is the number of generators and P_{Gi} is the power output of the *i*-th generator.

There may be discontinuities in the fuel cost curve. When the output power is expanded by utilizing additional boilers, steam condensers or other equipment those discontinuities occur. They may also occur if the cost represents the whole power station.

In the real ELD systems, input-output characteristics of the system display high nonlinearities and discontinuities because of valve-point loading in the plant. For enhancing the accuracy of the cost curve, valve-point effect is also included in the cost model. Thus, it takes the following form, also, Figure 2.3 shows the operating cost curve with valve point characteristics.

$$C_F(P_G) = \sum_{i=1}^{N} (a_i + b_i P_{Gi} + c_i P_{Gi}^2 + \left| e_i \sin(f_i (P_{Gi}^{\min} - P_{Gi})) \right|) \tag{2.2}$$

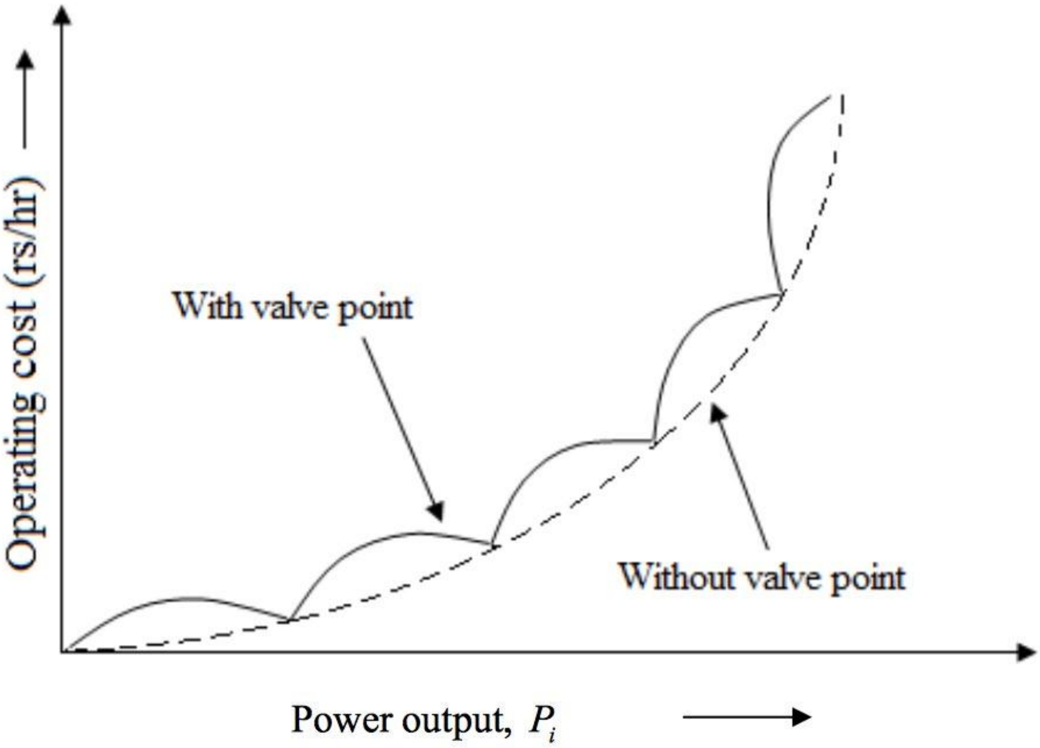


Figure 2.3 Operating cost curve with valve-point effects (Kaur, 2011)

where e_i and f_i are the constants modeling the valve-point effects for the *i*-th generator and $P_{Gi}^{\min}$ is the generator's minimum capacity. In electricity power generators, due to the choking of the partially open valves, modelling valve-point effects is a requirement to capture the losses aroused (Fraga, Yang & Papageorgiou, 2012). However, in this case, the number of local optima increases because of the ripples caused by the valve-point effect.

2.5 Emission Dispatch

EED problem deals with minimization of the emission of atmospheric pollutant gases that comes out of the burned fossil fuels. The amount of burned fossil fuel is in correlation with the amount of power generated at the unit. EED problem can be mathematically expressed, in ton/h, as,

$$C_E(P_G) = \sum_{i=1}^{N}(10^{-2}(\alpha_i + \beta_i P_{Gi} + \gamma_i P_{Gi}^2) + \zeta_i.\exp(\lambda_i P_{Gi})) \tag{2.3}$$

where $\alpha_i, \beta_i, \gamma_i, \zeta_i$ and λ_i are the emission coefficients for the *i*-th generation unit, values of those coefficients are given in Appendix B, and $C_E(P_G)$ is the total emission cost.

2.6 Constraints

In the solution process, we taken into account both equality and inequality constraints. The inequality constraint arises from power generation capacity, which is required for a stable operation. Each generation unit is limited to work in an operating zone that can be mathematically expressed as

$$P_{Gi}^{\min} \le P_{Gi} \le P_{Gi}^{\max} \qquad i = 1, 2, ..., N \tag{2.4}$$

Coming to equality constraints, one of them is due to power balance. Indeed, total power generation must be equal to the sum of the total demand and power losses occurring in transmission lines. The power balance constraint can be mathematically expressed as,

$$\sum_{i=1}^{N} P_{Gi} = P_D + P_L \tag{2.5}$$

where P_D and P_L are power demand and power losses, respectively. The other equality constraint, referring to power losses in transmission lines, is a function of the outputs of the generation units. Power losses can be written as follows:

$$P_L = \sum_{i=1}^{N} \sum_{j=1}^{N} P_{Gi} B_{ij} P_{Gj} + \sum_{i=1}^{N} B_{0i} P_i + B_{00} \tag{2.6}$$

where B_{00}, B_{0i} and B_{ij} are transmission loss coefficients.

2.7 System Variables and Problem of Optimum Dispatch

System variables are needed to know to analyze a power system network (Sivanagaraju & Sreenivasan, 2009). They are control variable, disturbance variable and state variable. Control variable is the real power generation P_G. It is used to control the state of the system. Disturbance (or demand) variable is the real power demands P_D. It is beyond the system control so it is considered as uncontrolled or disturbance variable. State variable is bus voltage magnitude *V*. It is a dependent variable that is being controlled by the control variables.

The process of allocation of generation among different generating units is called scheduling. Economic scheduling is a problem that allocating the generation among different generating units in such a way that overall cost of generation should be minimum. This problem is also called optimal dispatch.

Assume the total demand on the station as P_D and total number of generating units as *N*. The optimization problem is to allocate the total load P_D among these *N* different generation units, which all have different characteristics from each other, in an optimal way to reduce the overall cost of generation. Assume that P_{G1}, P_{G2}, P_{G3}, ..., P_{GN} are the powers generated by each individual unit to supply a load demand of P_D.

By optimizing total fuel cost function $C_F(P_G)$ and total emission cost function $C_E(P_G)$ subject to given equality and inequality constraints, control variable P_G is determined. Total of P_G must be equal to demand variable P_D. With the determined control variable and known cost and emission coefficients, total fuel cost $C_F(P_G)$ and emission cost $C_E(P_G)$ can be calculated.

2.8 The Multiobjective Economic Emission Load Dispatch Problem

2.8.1 Multiobjective Programming

Conventional parameter optimization problems try to find a single optimized solution based on a weighted sum of all objectives. If all objectives get better or worse at the same time together, optimal solution can be found effectively. However, if all objectives do not get better or worse at once, in other words they conflict with each other, then there are more than one optimal solution. In this case, a multiobjective study should be done which represents the tradeoff between the objectives. This study is called Pareto optimization. An example of pareto optimization is given in Figure 2.4.

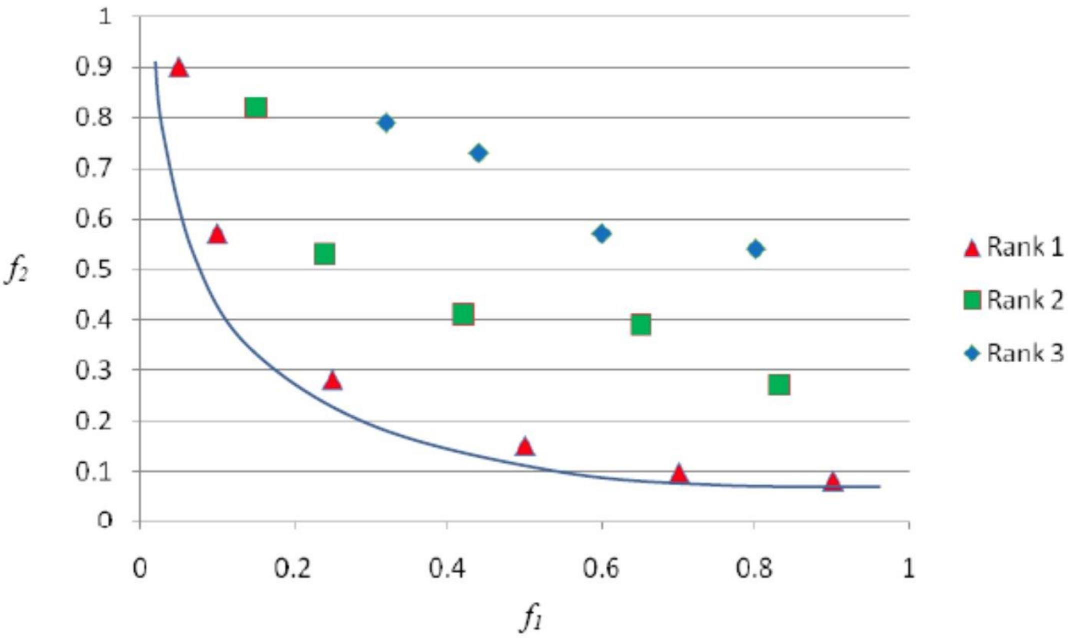


Figure 2.4 Results of an example pareto optimization study, where the conflict between f_1 and f_2 is explored. In this example, rank 1 is a nondominated set while rank 2 and rank 3 are higher determined. The true pareto front is drawn with solid blue line. (Weighted sum method, n.d.)

Mathematical definition of a multi-objective optimization problem is given as follows (Chase, Rademacher, Goodman, Averill & Sidhu, 2009), Minimize (or maximize):

$$f_i(x_1, x_2, ..., x_n), \qquad i = 1,2,...,p$$

such that:

$$h_j(x_1, x_2, ..., x_n) < 0, \qquad j = 1,2,...,q \tag{2.7}$$

where:

$(x_1, x_2, ..., x_n)$ are the n design variables

$f_i(x_1, x_2, ..., x_n)$ are the p objective functions

$h_j(x_1, x_2, ..., x_n)$ are the q inequality consraints

There are multiple Pareto optimal solutions for multiobjective optimization problems. Therefore, different researchers have proposed different ways to solve multiobjective optimization problems. Many methods actually convert the multiobjective optimization problem into a single objective optimization problem. This is called scalaratization problem. If scalarization is done carefully, pareto optimality of the solutions are almost guaranteed. (Multi-objective optimization, n.d.)

Most of the time, a human Decision Maker (DM) has to select the most appropriate pareto optimal set according to his/her preferences. The DM is expected to be an expert in this domain. Multiobjective optimization methods can be divided into four categories. These are; no DM (also known as no-preference) methods, a priori, a posteriori and interactive methods.

In a priori methods, a preference information is first asked from the DM, then a solution best satisfying to these preferences is found. In posteriori methods, candidate pareto optimal solutions are found and DM chooses one of them. In interactive methods, DM iteratively searches for the most preferred solution. DM is shown how to improve the solution for the next iteration. Therefore, a preferrable solution is found after few iterations.

2.8.1.1 Scalarizing Multiobjective Optimization Problems

Scalarizing a multiobjective optimization problem means formulating a single-objective optimization problem in a way that solution of that single-objective problem are pareto optimal solutions of the multi-objective optimization problem. With different parameters for the scalarization, different pareto optimal solutions are found. Mathematical formulation of the scalarization is described as follows,

$$\begin{aligned} &min \quad g\left(f_1(x),\ldots,f_k(x),\theta\right) \\ &s.t \; x \in X_\theta \end{aligned} \tag{2.8}$$

where θ is a vector parameter, the set $X_\theta \subseteq X$ is a set depending on the parameter θ.

One of the well known scalarization technique is call linear scalarization, also known as weighted sum method. Weighted sum method converts the multiobjective optimization problem into a scalar problem by constructing a weighted sum of all objectives. Linear scalarization can be mathematically expressed as,

$$min \sum_{i=1}^{k} w_i f_i(x) \tag{2.9}$$

This problem can be minimized by using a unconstrained optimization problem. A graphical illustration of linear scalarization is given in Figure 2.5,

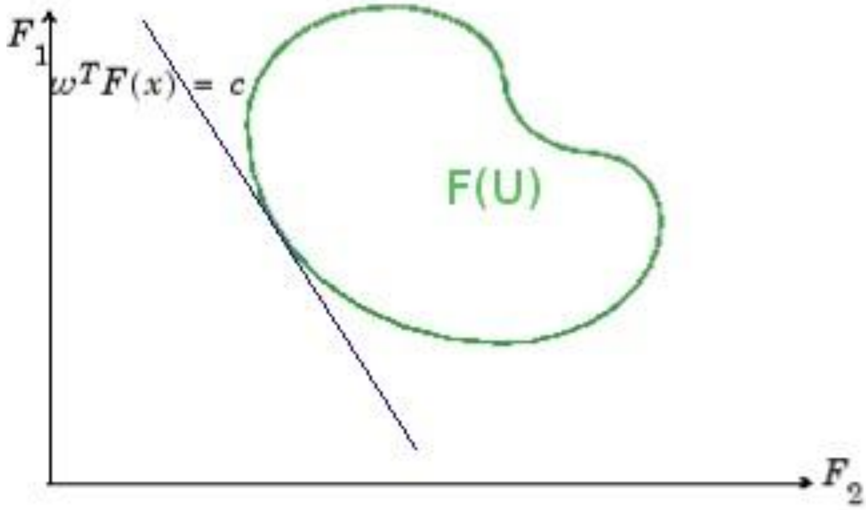


Figure 2.5 Graphical illustration of linear scalarization (Weighted sum method, n.d.)

In Figure 2.5, the objective function space a line, $w^T F(x) = c$. The minimization problem can be described as finding the value of c, at where the line touches the boundary of $F(u)$. Selection of weights defines the slope of the line, which leads to the true solution point.

2.8.1.2 No-Preference Methods

Multiobjective optimization methods that do not require a decision maker are called no-preference methods. A famous example is the method of global criterion, which can be mathematically expressed as,

$$\begin{aligned} &min\left\| f(x) - z^{ideal} \right\| \\ &s.t.\ x \in X \end{aligned} \tag{2.10}$$

where $\|.\|$ can be any L_p norm, with common choices including L_1, L_2 and L_∞.

2.8.1.3 A Priori Methods

A priori methods require sufficient priori information about the problem before the solution process begins. Well-known examples of a priori methods are the utility function method, lexicographic method and goal programming.

In the utility function method, it is assumed that the decision maker's utility function is available. A mapping is utility function if for all $y^1, y^2 \in Y$ it holds that $u\left(y^1\right) > u\left(y^2\right)$ if the decision maker prefers y^1 to y^2 and $u\left(y^1\right) = u\left(y^2\right)$ if the decision maker is indifferent between y^1 and y^2. Once u is found, we should solve,

$$max\ u\left(f\left(x\right)\right) \text{ subject to } x \in X \tag{2.11}$$

2.8.1.4 A Posteriori Methods

A posteriori methods aim to produce all Pareto solutions or a representetive subset of solutions. They can be grouped in two classes, mathematical programming-based methods, in which each run of the algorithm produces one Pareto optimal solution and evolutionary algorithms, in which each run of the algorithm produces a set of Pareto optimal solutions.

Well-known examples to mathematical programming –based methods are Normal Boundary Intersection (NBI) and Directed Search Domain (DSD). They try to solve the multiobjective problem by constructing several scalarizations. The solution of each scalarization leads to a global or local Pareto optimal solution.

Evolutionary algorithms are more popular approaches when solving a multiobjective optimization problem. Most of them apply Pareto-based ranking schemes. Some well-known examples are Non-dominated Sorting Genetic Algorithm-II (NSGA-II) and Strength Pareto Evolutionary Algorithm 2 (SPEA 2). The main advantage of these algorithms are they produce a whole set of Pareto optimal solutions each run, however, they are slower and Pareto optimality of the solution is not guaranteed.

2.8.1.5 Interactive Methods

In interactive methods, the soution process is iterative and the DM is required to take a decision at each iteration to obtain a Pareto optimal solution. Steps an interactive method is given below.

1. Initialize the problem (calculate the approximated and ideal objective vectors and show them to the DM)
2. Generate a Pareto optimal starting point (by the solution given by the DM or some no-preference method)
3. Ask for preference information from the DM (number of new solutions to be generated or aspiration levels)
4. Generate a new set of Pareto optimal solutions according to the preferences and show the information to the DM.
5. If several solutions are generated, ask the DM which is the best one.
6. Stop if the DM wants, otherwise, go to step 3.

2.8.2 Solution of the Multiobjective EELD problem with WSM Method

The EELD problem aims at finding a balance between the total fuel cost (ELD problem) and the emission (EED problem). Therefore, it is a multi-objective optimization problem and mathematically it can be formulated as follows,

$$\text{minimize } C_F(P_G) \text{ and } C_E(P_G) \tag{2.12}$$

subject to

$$P_{Gi}^{\min} \le P_{Gi} \le P_{Gi}^{\max} \qquad i = 1,2,...,N \tag{2.13}$$

$$\sum_{i=1}^{N} P_{Gi} = P_D + P_L \tag{2.14}$$

$$P_L = \sum_{i=1}^{N}\sum_{j=1}^{N} P_{Gi} B_{ij} P_{Gj} + \sum_{i=1}^{N} B_{0i} P_i + B_{00} \tag{2.15}$$

In this thesis, we tackle Equations (2.12), (2.13), (2.14) and (2.15) by proposing a new, efficient and effective heuristic optimization algorithm. The main difficulty with the problem is that it is necessary to find best compromising solutions which minimize C_F and C_E, subject to the given constraints. For this, we use pareto-based approach. A group of best compromising solutions is said to be pareto-optimal if an improvement in one objective leads to impairment of another objective. In the present thesis, to extract the best-compromised solutions out of a group of pareto-optimal solutions, the WSM method (Rajasomashekar & Aravindhababu, 2012) is used to express the problem as a single-objective optimization problem. Mathematical expression of the WSM reads as:

$$\text{minimize} \sum_{i=1}^{N} \left(wC_F\left(P_{Gi}\right) + \left(1-w\right) hC_E\left(P_{Gi}\right) \right) \tag{2.16}$$

where h is the price penalty factor, which combines the fuel cost with the emission cost to obtain the total cost of the system. And w is the weight parameter of the pareto-curve in the range of [0, 1]. When w equals 1, the problem turns into a single-objective ELD problem pertaining only to minimization of the fuel cost. Between the values of 0 and 1, w blends both EED and ELD problems to form an optimum solution. By increasing the value of w from 0 to 1, the significance of emission in the problem increases while the significance of the fuel cost decreases. When w equals zero, the problem turns into a single objective EED problem, which deals with the minimization of just the emission.

The constrained optimization problem in Eq. (2.12), along with the constraints Eq. (2.13), Eq. (2.14) and Eq. (2.15) can be solved for different values of w and a pareto curve with non-dominated solutions is obtained. However, this approach may not yield the best compromising solution, which is presented as the equal percentage weights of the optimal solutions of the ELD and EED problems. So w can be set to 0.5 to obtain a desirable best compromising solution, if the chosen h parameter value makes both fuel cost and emission to the same level in the objective function. The price penalty factor h is the ratio between maximum fuel cost and maximum emission of the corresponding generator. In the literature, methods to calculate approximate values of the *h* parameter are introduced (Venkatesh, Gnanadass & Padhy, 2003). They, however, do not always lead to good results. Therefore, one possibility is to eliminate the *h* parameter from the cost function as we will prefer to do. Accordingly, we adopt the method of Rajasomashekar and Aravindhababu (2012) whose approach involves normalizing both fuel cost and emission components by ascribing equal significances to both objectives as,

$$\text{minimize } w\left[\frac{\sum_{i=1}^{N} C_{Fi}(P_{Gi}) - C_F^{\min}}{C_F^{\max} - C_F^{\min}}\right] + (1-w)\left[\frac{\sum_{i=1}^{N} C_{Ei}(P_{Gi}) - C_E^{\min}}{C_E^{\max} - C_E^{\min}}\right] \tag{2.17}$$

where $C_F^{\min}, C_F^{\max}, C_E^{\max}$ and $C_E^{\min}$ are maximum and minimum weighted values of fuel cost and emission cost in the multi-objective optimization problem, obtained by solving Eq. (2.12) for both single-objective ELD and EED problems. Obviously, Eq. (2.17) avoids use of the h parameter.

CHAPTER THREE
OPTIMIZATION ALGORITHMS

3.1 Optimization

Most optimization problems can be mathematically expressed as follows (Yang, 2010):

$$\underset{x \in \Re^n}{\text{minimize}} \ f_i(x), \qquad (i = 1,2,...,M), \tag{3.1}$$

$$\text{s.t.} \ \phi_j(x) = 0, \qquad (j = 1,...,J), \tag{3.2}$$

$$\psi_k(x) \leq 0, \qquad (k = 1,...,K), \tag{3.3}$$

where $f_i(x)$, $\phi_j(x)$ and $\psi_k(x)$ are functions of the design vector

$$x = (x_1, x_2, ..., x_n)^T . \tag{3.4}$$

where the components x_i of x are called decision or design variables, and they can be continuous, discrete or the mix of these two. The function $f_i(x)$ where $i = 1,2,...,M$ are called objective functions. In the case of $M = 1$, it is a single objective function, if $M > 1$, then it is called multiobjective function. It is sometimes called cost or energy function in the literature. The space spanned by the decision variables is symbolized with $\Re^n$ and it is called the search or solution space. Finally, the equalities for ϕ_j and inequalities for ψ_k are called constraints.

In the case of there are not any objective functions, but only constraints, it is called feasibility problem because any feasible solution is an optimal solution.

Sometimes, it is not possible to write the objective function in the explicit form. Such as, if we want to design a car engine, we may want to design the engine with highest fuel efficiency and lowest carbon dioxide emission. However, these objectives depend on various factors, such as, the type of fuel, ignition system and geometry of the engine. Therefore, we may want to use different tools to investigate further this relationship, for example computational fluid dynamics (CFD) softwares, however, the relationship between the efficiency and the factors may still be complex, thus no explicit form is possible. In this case, we are dealing with the black-box type optimization problem and they are difficult to solve.

In some other cases, the objective function may not be measurable, but we want to still maximize it. Such as, when we go on holiday, we may want to maximize our enjoyment but minimize the costs. This is a difficult problem and it is not easy to express this problem with mathematical terms because level of enjoyment may differ from people to people. Mathematically, we can only minimize or maximize something if it can be expressed in mathematical terms.

Also, the functions f_i, ϕ_j and ψ_k may include integrals that make things complex. In this case, most of the time we have to use calculus of variations. For example, if we want determine the shape of a hanging rope anchored with two points A and B, we have to calculate a shape $y(x)$ which makes the total energy E_p minimal.

$$\underset{y(x)}{\text{minimize}} \quad E_p = \rho g \int_A^B y\sqrt{1+y'^2}\,dx \tag{3.5}$$

where g is the gravitational acceleration and ρ is the mass per unit of the rope. Furthermore, length of the rope is fixed which creates the following constraint,

$$\int_A^B \sqrt{1+y'^2}\,dx = L \tag{3.6}$$

These optimization problems requires to solution of the Euler-Lagrange equation. Also, if the objective function include integrals and constraints are expressed in terms of differential equations, these problems are called optimal control problem, which is a common control technique.

3.2 Optimization Algorithms

A classification of optimization algorithms is given in Figure 3.1.

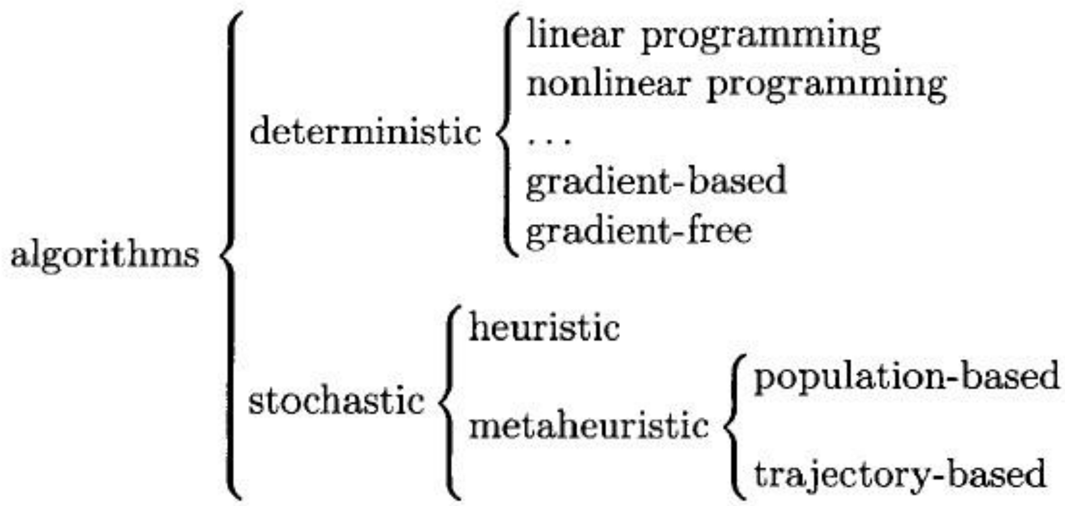


Figure 3.1 Classification of optimization algorithms (Yang, 2010)

We often use different kind of optimization algorithms for different kind of problems. We can think the optimal solution search process as trying to hunt for a hidden treasure in a hilly landscape within a time limit. Furthermore, we can also suppose that we are blind-folded, therefore this search process is random. If we are told that the treasure is at the peak of a known region, then we will climb up to the cliff to reach that peak and this scenario corresponds to the classical hill-climbing techniques.

When we are not blind-folded and we do not know where to look. Looking at every centimeter square of an extremely hilly large region is an absurd idea. Therefore, we search an area randomly looking for a plausible place, collect some hints from that region and then search another area with random walks and so on. Such random search is the basis of today’s modern search methods. Also, we can either do the treasure-hunting alone, which is named as trajectory-based search (i.e. Simulated Annealing algorithm), or with a group while each agent both hunting and share the information they gathered with each other. The latter scenario is called swarm intelligence and if the area is extremely large, then these kind of algorithms are preferable.

Also, we can change our search strategy a little. Most of the time, some hunters are better than others. So, keeping the better hunters and recruiting the new ones is a good strategy. This strategy is used in many modern metaheuristic algorithms.

Generally, optimization algorithms can be divided into two categories: stochastic algorithms and deterministic algorithms. Deterministic algorithms follow a rigorous procedure and repeatable. For instance, if we run the hill-climbing algorithm, which is a deterministic algorithm, it will follow the same path whether we run the program today or tomorrow. On the other hand, stochastic algorithms have some randomness. Genetic algorithms are a good example of stochastic algorithms. Each run of the program, although the solution may not be very different, the strings or solutions in the population will be different. Also, there is a third type of algorithm, which is the mixture of deterministic and stochastic algorithms. Hill-climbing problem with a random starting point is a good example. These algorithms are not repeatable and offer a solution to stuck in a local optima problem.

Most classic algorithms are deterministic. For instance, the Simplex method is deterministic. Deterministic methods can be divided into two groups: gradient-based and non-gradient-based algorithms. For instance, the Newton-Raphson method is a gradient-based algorithm, which uses the function values and their derivatives to solve the optimum point. It works with smooth unimodal problems, however, if there is a discontinuity in the problem, it does not work well. Non-gradient-based methods do not use derivative information but only function values and they are preferable when there is a discontinuity in the function. Nelder-Mead downhill simplex and Hookes-Jeeves pattern search are examples of the non-gradient-based algorithms.

Stochastic algorithms can be divided into two groups: heuristic algorithms and metaheuristic algorithms. Heuristic means ‘to discover by trial and error’ or ‘to find’. These algorithms are expected to work most of the time, but not all the time. Heuristic algorithms are preferred when we do not necessarily wish the best solution but rather a good solution.

Metaheuristic algorithms extend the heuristic algorithms and most of the times perform better. Meta- means ‘higher level’ or ‘beyond’. Furthermore, all metaheuristic use certain local search and tradeoff of randomization. Randomization is a good option when performing a global search. Therefore, metaheuristic algorithms are suitable for global optimization.

3.3 Metaheuristics

Most metaheuristic algorithms are inspired from nature. Nature has found perfect solutions to almost all problems she came across for over millions of years. Therefore, we can inspire from the nature and apply her problem solving techniques to the optimization problems. Some nature-inspired algorithms are based on Darwin’s evolutionary theory. Therefore, we can say that they are biology-inspired.

Two most important aspects of the metaheuristic algorithms are randomization and selection of the best solutions. Randomization ensures that the algorithm does not stuck into local optima while selection of the best solution ensures that the solutions will converge optimally.

One way to classify metaheuristic algorithms are trajectory-based algorithms and population-based algorithms. Genetic algorithms use a set of strings so we can say that they are population-based so is the particle swarm optimization, which uses particles.

Simulated Annealing algorithm, which uses a single agent, is an example for trajectory-based algorithms. In the Simulated Annealing algorithm, the agent searches the design space in a piecewise style. The moves follow a trajectory in the design space, with a non-zero probability of finding the optimal solution.

3.3.1 History of Metaheuristic Algorithms

Heuristics is a trial-and-error solution strategy to find acceptable solutions to an optimization problem in a reasonably short time. The complexity of the problem makes it impossible to search for every possible solution or combination, with the heuristic optimization algorithms, feasible solution can be acquired in a reasonably practical time. There is no guarantee that the algorithm will find the best solution or even the algorithm will work. The idea is based on producing good quality solutions with a practical and efficient algorithm.

Alan Turing was maybe the first to use the heuristic algorithms during the Second World War when he was breaking the German Enigma codes. He designed an electromechanical machine, the Bombe, to help the code breaking work. The Bombe used an heuristic algorithm, with Turing's words, searched for 10^{22} possible combinations to find the correct message in an Enigma code. Turing called his search tecnique heuristic search, although he was not sure if the technique will work or not, it became a huge success. In 1945, Turing was recruited to the National Physical Laboratory (UK), where he published most of his innovative in machine intelligence and neural networks, also an early form of genetic algorithms.

The next step in the development of evolutionary algorithms was John Holland and his collaborators in University of Michigan developed the Genetic Algorithms (GA) in 1960s and 1970s. GAs is a natural abstraction of Darwin's works in evolution and natural selection of biological systems. Ever since the development, GAs have become a huge success. Even Fortune 500 companies are still using them in their routine optimization tasks such as planning, data-fitting and scheduling.

In 1992, Marco Dorigo finished his PhD thesis on optimization and natural algorithms, in which he described an innovative study on Ant Colony Optimization

(ACO). It was inspired by the social intelligence of ants using a chemical as a chemical messenger.

In the 21st century, things became even more exciting. First, Zong Woo Geem et al. in 2001 developed the Harmony Search algorithm, which has been widely accepted in the literature as solving optimization problems such as water distribution, transport modelling and scheduling. Then, in 2005, D. Pham et. al. published their work on Bee Colony algorithm and D. Karaboga published the Artificial Bee Colony (ABC) algorithm.

As can be seen more and more metaheuristic algorithms are still developed. This thesis deals with a hybrid Artificial Cooperative Search – Quadratic Approximation algorithm. As an example, Particle swarm optimization will be discussed in the following section.

3.3.2 Particle Swarm Optimization

Researchers have been observing collective behavior of natural systems for years. Intelligence does not reside in an individual but rather distributed among a group of individuals in such systems. These behaviors can be seen in swarm of animals when they seek food, avoid predators or travel more quickly.

Animal groups more often avoid predators when they are in a group. For instance, when a zebra is alone, it may be easy for a lion to notice the zebra due to its' contrast with the surrounding landscape, however, a group of zebras can blend together and it may be hard to notice the individuals. Also, a group can more effectively feed in an area when they form a group. For instance, when they are drinking from a stream, random effects dictate that there will be always a few animals, who are watching around for predators.

Particle swarm optimization (PSO) is based on these observations that when a group of individuals work together they improve not just their collective performance, but also each individual performance

Suppose that we have a minimization problem, which is defined over a continuous domain of d dimensions. We have also a population of N candidate solutions, shown as $\{x_i\}, i \in [1, N]$. Also, assume that each particle, or individual, x_i moves in the search space with some velocity v_i. This movement in the search space is the strength of the PSO algorithm. Other Evolutionary Algorithms (EAs) are more static than PSO because they model the evolution from one generation to the next, while PSO models the dynamic behavior of the particles in a search space.

As a PSO individuals moves through the space, it has some inertia to maintain its' velocity. However, its' velocity may change due to a couple reasons:

- First, if it remembers its' best position in the past and decides to move back to that position. It changes its' velocity.
- Second, an individual knows the best position of the neighboring particles at the current generation. This requires the definition of a neighborhood size, and it requires the communication of all particles between each other.

The basic PSO algorithm can be summarized as in Figure 3.2,

Initialize a random population of individuals $\{x_i\}$, $i \in [1, N]$
Initialize each individual's n-element velocity vector v_i, $i \in [1, N]$
Initialize the best-so-far position of each individual: $b_i \leftarrow x_i$, $i \in [1, N]$
Define the neighborhood size $\sigma < N$
Define the maximum influence values $\phi_{1,\max}$ and $\phi_{2,\max}$
Define the maximum velocity $v_{\max}$
While not(termination criterion)
    For each individual x_i, $i \in [1, N]$
        $H_i \leftarrow \{\sigma$ nearest neighbors of $x_i\}$
        $h_i \leftarrow \arg\min_x\{f(x) : x \in H_i\}$
        Generate a random vector ϕ_1 with $\phi_1(k) \sim U[0, \phi_{1,\max}]$ for $k \in [1, n]$
        Generate a random vector ϕ_2 with $\phi_2(k) \sim U[0, \phi_{2,\max}]$ for $k \in [1, n]$
        $v_i \leftarrow v_i + \phi_1 \circ (b_i - x_i) + \phi_2 \circ (h_i - x_i)$
        If $|v_i| > v_{\max}$ then
            $v_i \leftarrow v_i v_{\max} / |v_i|$
        End if
        $x_i \leftarrow x_i + v_i$
        $b_i \leftarrow \arg\min\{f(x_i), f(b_i)\}$
    Next individual
Next generation

Figure 3.2 A basic particle swarm optimization algorithm. (Simon, 2013)

Some tips while tuning the parameters of the PSO algorithm are:

- Like the other EAs, we have to initialize the population and corresponding velocity vectors.
- We have to define the neighborhood size σ of the algorithm. The term "neighborhood size" is ambiguous. Sometimes it means each particle in the swarm has σ close neighbors, while sometimes each particle has $(\sigma - 1)$ close neighbors.
- We have to choose the parameters ϕ_1 and ϕ_2. The parameter ϕ_1 is called the cognition learning rate, and ϕ_2 is called the social learning rate. Most of the time $\phi_{1,max}$ and $\phi_{2,max}$ are often chosen as 2.05.
- We have to define a maximum velocity v_{max} for each individual. Which is understandable, if the v_{max} is greater than the search space, than a particle can easily leave the search space in a single generation.
- Velocity can be updated with the following formula,

$$v_i \leftarrow v_i + \phi_1 (b_i - x_i) + \phi_2 (h_i - x_i) \tag{3.7}$$

where ϕ_1 and ϕ_2 are scalars with ranges $\phi_1 : U\left[0,\phi_{1,max}\right]$ and $\phi_2 : U\left[0,\phi_{2,max}\right]$. This option is called linear PSO and each element of the velocity vector v_i is updated with the same values of ϕ_1 and ϕ_2.

- Like the other EAs, elitism improves the performance of PSO.
- The update equation $x_i \leftarrow x_i + v_i$ may result in x_i moving outside the search domain. We generally use limit x_i to keep it in the search domain. For example, we can use the following formulas,

$$\begin{aligned} x_i &\leftarrow min\left(x_i, x_{max}\right) \\ x_i &\leftarrow max\left(x_i, x_{min}\right) \end{aligned} \tag{3.8}$$

where $\left[x_{min}, x_{max}\right]$ defines the limits of the search domain.

3.4 Types of Optimization Problems

A classification list of optimization problems is given in Figure 3.3.

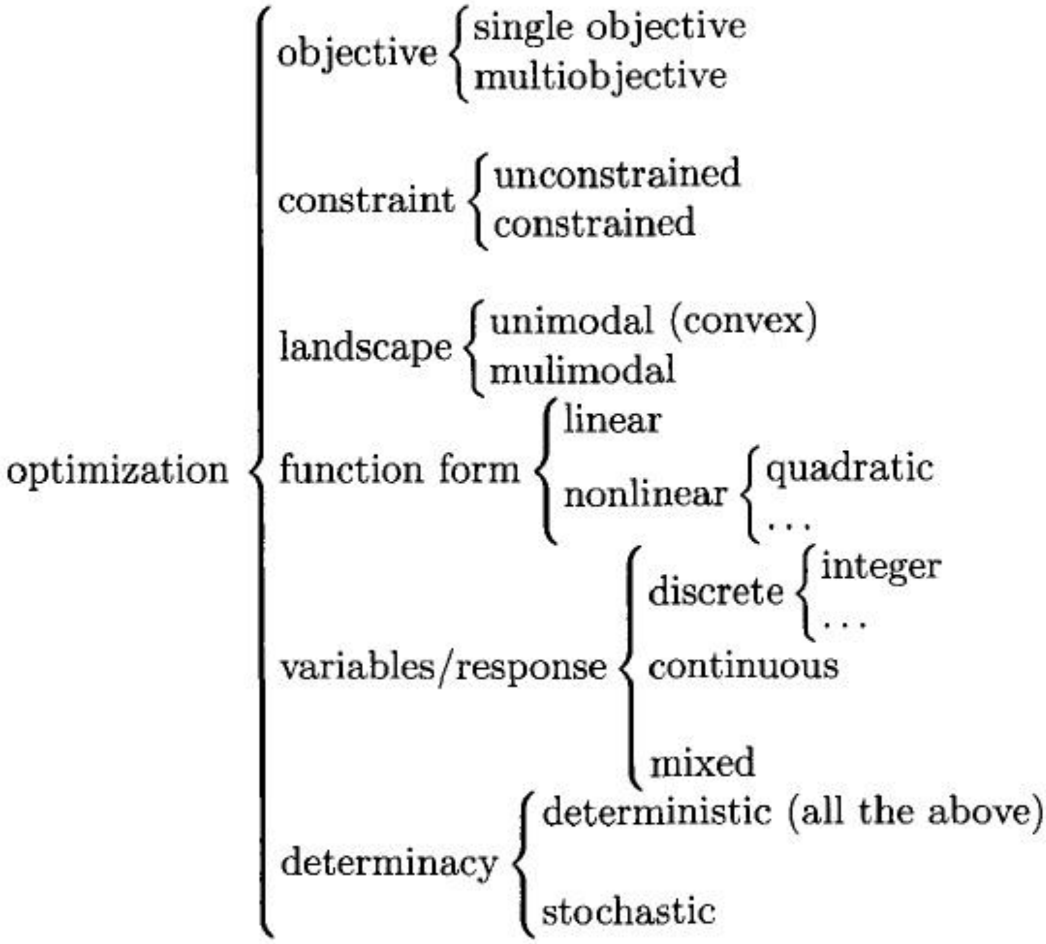


Figure 3.3 Classification of optimization problems (Yang, 2010)

If we decide to classify optimization problems by their number of objectives, we can find two kinds: single objective ($M = 1$) and multiobjective ($M > 1$). Multiobjective optimization problems are also known as multicriteria or multi-attributes optimization problems. In the real world, most optimization problems are multiobjective. For instance, if we want to design a car engine, we may want to minimize carbon-dioxide emission, maximize fuel efficiency and lower its noise level. There are three objectives in this problem and we need some compromise between the objectives since the objectives are often conflicting with each other. A good example of multiobjective optimization problems can be the following problem (Simon, 2013);

$$\min_x \left[f(x) \text{ and } g(x) \right], \quad \begin{array}{ll} \text{where} & f(x) = x^4 + 5x^3 + 4x^2 - 4x + 1 \\ \text{and} & g(x) = 2(x+1)^2 . \end{array} \tag{3.9}$$

In this problem, we want to minimize $f(x)$ and $g(x)$ at the same time, which makes the problem multiobjective. Graphical representation of the algorithm is given in Figure 3.4.

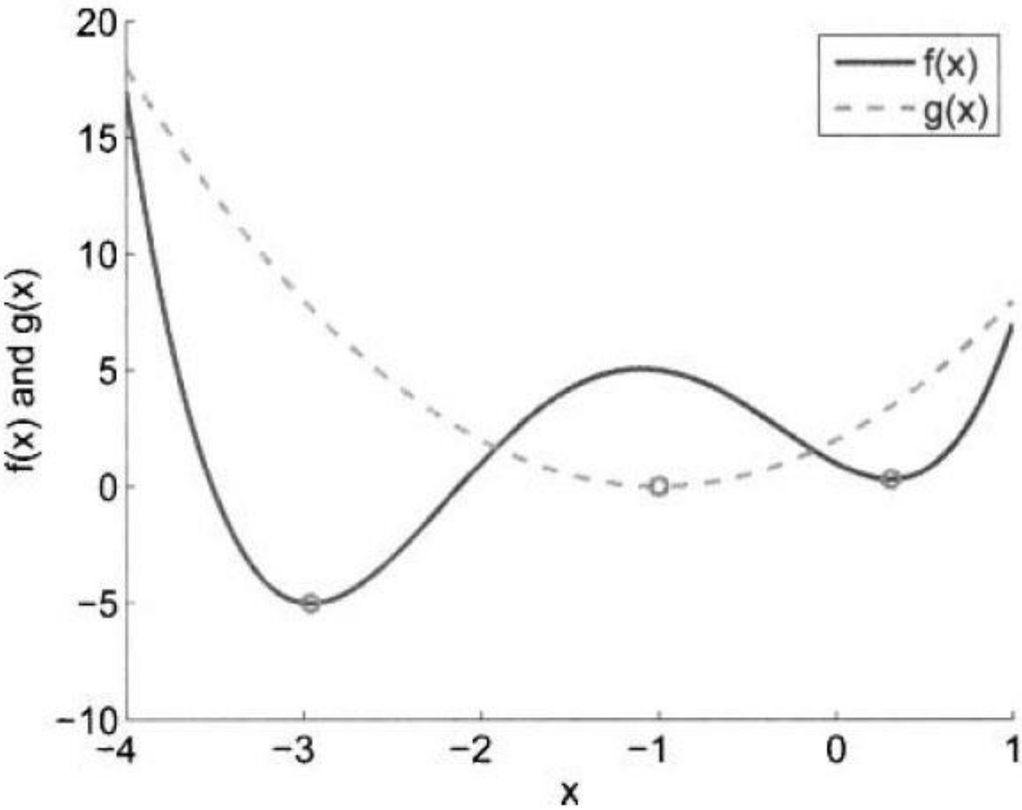


Figure 3.4 A simple multiobjective optimization problem where two objective conflict. (Simon,2013)

As can be seen in Figure 3.4, $x = -2.96$ minimizes $f(x)$, $x = -1$ minimizes $g(x)$. However, we do not know the x value which minimizes $f(x)$ and $g(x)$ at the same time. One way to solve this simple multiobjective optimization problem is to plot $g(x)$ as a function of $f(x)$. Plot of $g(x)$ as the function of $f(x)$ can be seen in Figure 3.5.

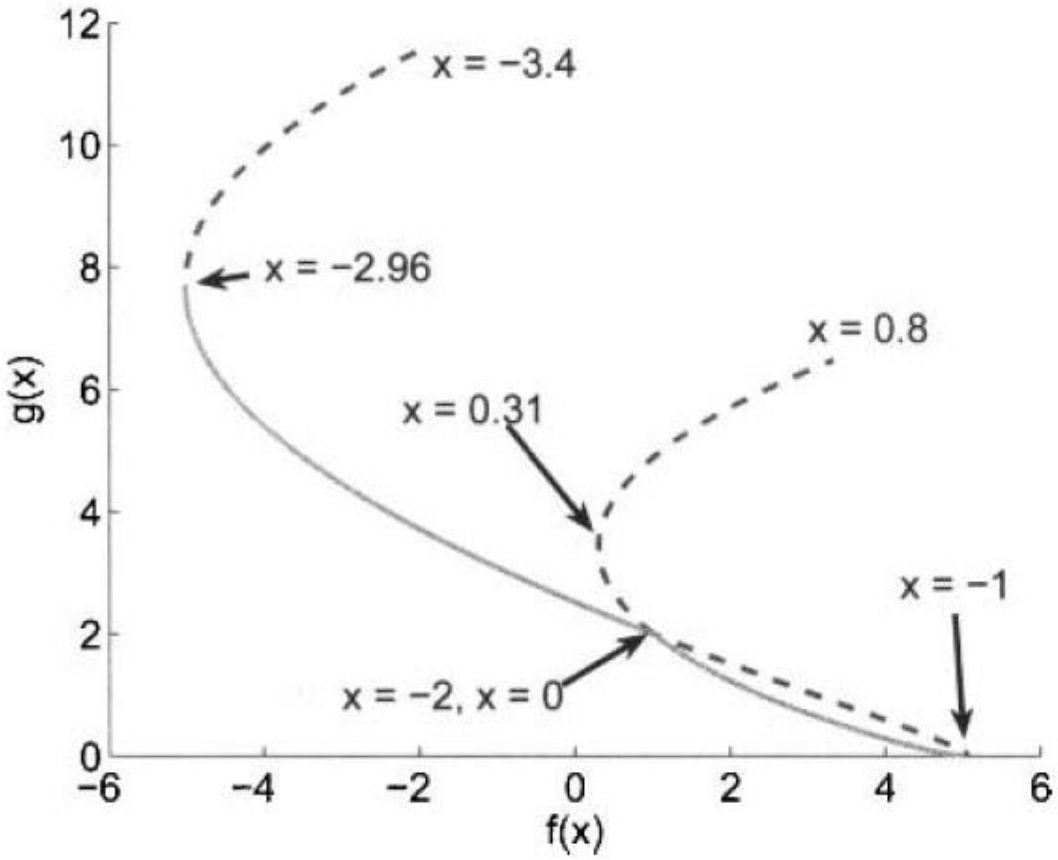


Figure 3.5 The plot of $g(x)$ as the function of $f(x)$ as x varies from -3.4 to 0.8. The solid line is the Pareto front. (Simon, 2013)

As we can see in Figure 3.5, between -2.96 and -1, $f(x)$ increases while $g(x)$ decreases. So, Pareto front must be at this interval. It is up to the person to select a point on the Pareto front. Every point on the Pareto front symbolizes a tradeoff between the objectives. We can also classify the optimization problems in terms of number of constraints $J+K$. If $J=0$ and $K=0$, this means there are no constraints at all in the problem. These kinds of optimization problems are called unconstrained optimization problems. If $J=0$ and $K\geq 1$, these problems are called inequality-constrained problem. If $K=0$ and $J\geq 1$, they are called equality-constrained problem. An example to unconstrained problems can be the following problem,

$$\min_{x} f(x), \text{ where } f(x) = x^4 + 5x^3 + 4x^2 - 4x + 1 \tag{3.10}$$

Also, plot of the function is given in Figure 3.6.

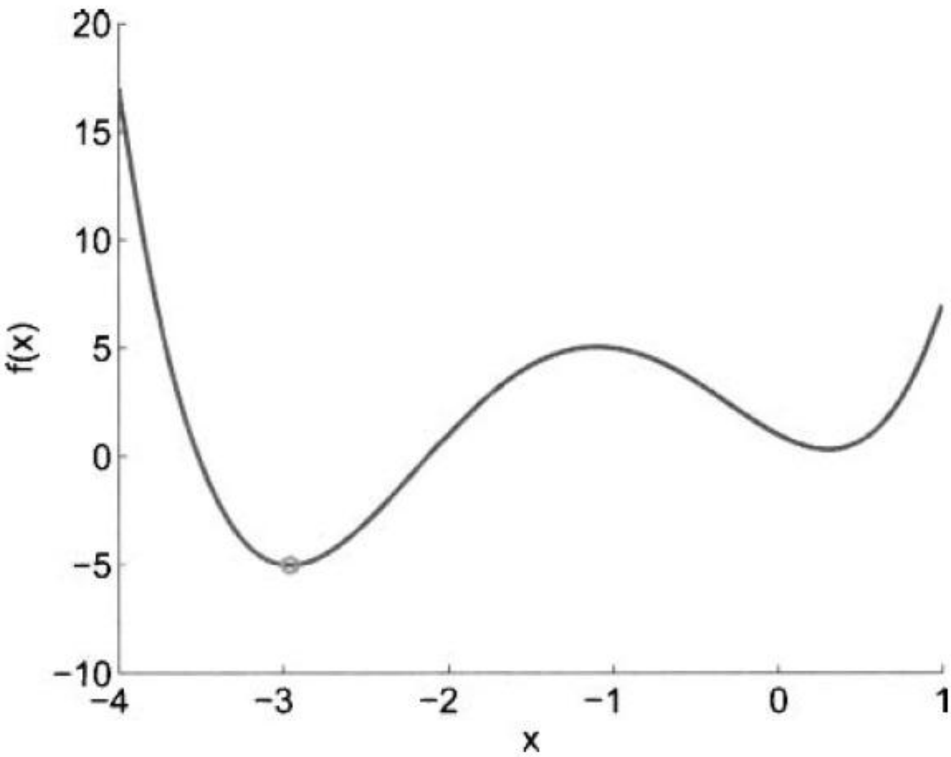


Figure 3.6 A simple unconstrained minimization problem (Simon, 2013)

The function is quadratic polynomial; therefore, we can say that it has at most three stationary points. We can find these points by calculating x at $f'(x) = 0$. These points are $x = -2.96$, $x = -1.10$ and $x = 0.31$. Furthermore, we can calculate the value of these points at $f''(x)$ and conclude that $x = -2.96$ is the global minimum.

An example to constrained optimization problems can be the following problem,

$$\min_{x} f(x) \quad \text{where} \quad f(x) = x^4 + 5x^3 + 4x^2 - 4x + 1 \qquad (3.11)$$
$$\text{and} \quad x \geq -1.5$$

Plot of this problem can be shown as in Figure 3.7.

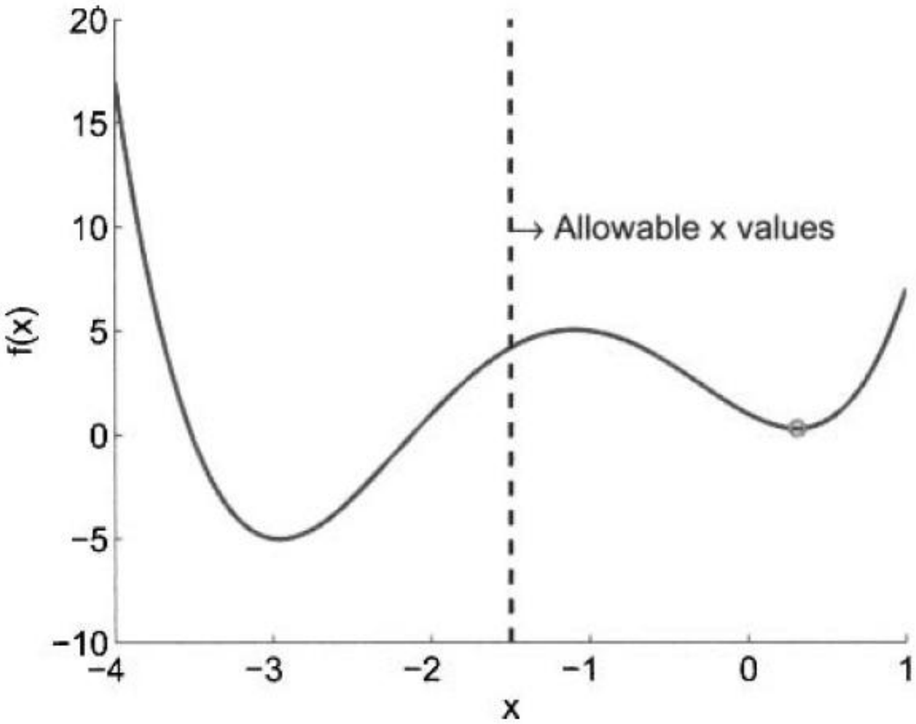


Figure 3.7 A simple constrained optimization problem. (Simon, 2013)

By looking at the last example's stationary points, we can see that only $x = 0.31$ satisfies the constrained. Thus, we can say that this points is the global minimum.

We can also classify the optimization problems as linear and nonlinear optimization problems. If the constraints ϕ_j and ψ_k are all linear, then we can say that it is a linearly constrained problem. If the constraints and objective functions are all linear, then it is a linear programming problem. If the constraints and objective functions are all nonlinear, then it is a nonlinear optimization problem.

From a different point of view, we can use the optimization problems in terms of the landscape of the objective functions. For a single objective function, if there is only one unique global optimum, then it is a unimodal problem. If there are more than one optimum points, then it is a unimodal problem. For example, $f\left(x,y\right)=x^2+y^2$ is unimodal problem, while $f\left(x,y\right)=sin\left(x\right)sin\left(y\right)$ is a multimodal problem.

Also, if the design variables are only discrete, we call this as discrete programming problem. If the design variables are integers, then we call this as integer programming problem. If the design variables are all continuous real values, then we call this as continuous optimization problem. If some design variables are integers and some of them are real values, then we call this as mixed type optimization problem.

Finally, we can classify the optimization problems as discrete and stochastic optimization problems. Practically, almost all optimization problems are deterministic. The design variables, constraint functions and objective functions are all determined exactly. However, in the real world, there may be some uncertainty in our knowledge of some parameters. For instance, when measuring the material properties of a product, Young’s modulus can be measured with certain accuracy with some uncertainty due to inhomogenities in the product. If there are uncertainties in the problem, we call this stochastic optimization problem. If all properties of the problem are determined exactly, then we call this as deterministic optimization problem.

3.5 Constraint Handling and Penalty Method

For a simple function with constraints, as the problem described in the thesis, several methods can be used to handle them. One of the most common, also used in this thesis, is the penalty method. It can be mathematically described as follows. For the optimization problem,

$$\begin{aligned}&\text{minimize } f(x),\ x=(x_1,x_2,...,x_N)^T,\\&\text{s.t. } \phi_i(x)=0,\ (i=1,...,M),\\&\quad \psi_j(x)\le 0,\ (j=1,...,N).\end{aligned} \tag{3.12}$$

the idea is to define a function that transforms the constrained problem into a unconstrained problem. Now we define,

$$\Pi(x,\mu_i,\nu_j)=f(x)+\sum_{i=1}^{M}\mu_i\phi_i^2(x)+\sum_{i=1}^{N}\nu_j\psi_j^2(x) \tag{3.13}$$

where $\mu_i >> 1$ and $\nu_j \ge 0$ should be large enough depending on the needed solution quality.

3.6 Artificial Cooperative Search Algorithm

The proposed algorithm hybridizes the ACS and QA to form an efficient algorithm to solve EELD problem. Therefore, we start discussing the algorithm by first introducing the ACS and QA algorithms

Artificial Cooperative Search (ACS), first introduced by Civicioglu (2013), is a swarm intelligence-based metaheuristic algorithm for solving numerical optimization problems. In nature, living species interact with each other through different kinds of relationships, such as predator-prey or parasite-host relationships. Cooperation and mutualism between two superorganisms living in the same habitat has inspired ACS algorithm. In ACS algorithm, two artificial superorganisms (namely predator and prey) migrate and interact with each other in a habitat to converge the global minimum of the optimization problem.

In nature, the amount of food in a specific habitat depends on seasonal climate changes. Thus, superorganisms develop seasonal migration behaviors to move more productive habitats from the habitats experiencing reduction in resources. Prior to migration, members of a species get together and form a superorganism. It is not known how members of a superorganism collectively decide on the direction and time of the migration. Generally, researchers explain the movement of a superorganism by using random-motion movement models (Civicioglu & Besdok, 2013). Also, prior to migration, superorganisms divide into sub-groups (sub-superorganisms) in such a way that decision of a superorganism is made by the coordination of sub-superorganisms. Superorganisms use explorers to discover new possible migration areas that contain relatively more resources. Explorer shares the data about the discovered areas with the superorganism. If superorganism decides that one of the possible migration areas is suitable for feeding and nesting, it moves to that area. Meanwhile, it continues finding more productive areas and migrates again.

In ACS algorithm, a superorganism is represented by an artificial superorganism which migrates to more productive areas as the search progresses. ACS algorithm contains two superorganisms, namely α and β, which consist of artificial sub-superorganisms equaling in number to the dimension of the population (N). The numbers of individuals within the sub-superorganisms are equal to the dimension of the problem (D). The two superorganisms, α and β, are used to determine predator and prey sub-superorganisms. In ACS algorithm, the predator sub-superorganism can trace prey sub-superorganism for a timeframe while they migrate to the global minimum of the problem. In the iterative part of the ACS algorithm, namely the coevolution process, two superorganisms look for the global minimum of the problem and establish a cooperative interaction with each other. Initial values of the individuals of the *i*-th sub-superorganism of α (i.e., $\alpha_{i,j}$) and β (i.e., $\beta_{i,j}$) can be defined as follows,

$$\begin{aligned} \alpha_{i,j:g} &= r.(u_j - l_j) + l_j \\ \beta_{i,j:g} &= r.(u_j - l_j) + l_j \end{aligned} \tag{3.14}$$

where i = 1, 2, 3,…, N, j = 1, 2, 3,…, D and g = 1, 2, 3,…, *maxcycle*. Here, g labels the generation number indicating the coevolution level of the related superorganism, and *maxcycle* stands for the maximum number of generations. r represents a random number taken from a uniform distribution within [0,1]. The u_j and l_j represent respectively the upper and lower limits of the search space for the j-th dimension of the problem. The fitness values f (*i.e.*, productivity values) for the corresponding sub-superorganism are calculated as,

$$\begin{aligned} y_{i;\alpha} &= f(\alpha_i) \\ y_{i;\beta} &= f(\beta_i) \end{aligned} \qquad (3.15)$$

The biological interaction location x between the prey and predator organisms can be modeled as,

$$x = \mathrm{P}_{red} + R.(\mathrm{P}_{rey} - \mathrm{P}_{red}) \qquad (3.16)$$

where R is the scale factor that controls the speed of the biological interaction and P_{red} and P_{rey} represent the predator and prey sub-superorganisms, respectively. In ACS algorithm, if the source of prey and predator sub-superorganisms selected randomly are the same as each other in the present generation then this means a self-interaction process for the predator. Also, due to the stochastic nature of the ACS algorithm, predator and prey roles of the sub-superorganisms can change in any generation. Thus, ACS algorithm provides a cooperative/coevolution process for both of the superorganisms. Pseudo code of the algorithm is given in Figure 3.8.

```
Data: N, D, maxcycle, low, up, f, p
Result: GlobalMinimizer | GlobalMinimum = f(GlobalMinimizer)
Superorganisms : α, β
// Initialization
GlobalMinimum_{g=0} ≈ ∞
for i ← 1 to N do
    for j ← 1 to D do
        α_{i,j}, β_{i,j} ~ U(low_j, up_j)
    end
    y_{i;α} = f(α_i)
    y_{i;β} = f(β_i)
end
for g ← 1 to maxcycle do
    // Selection
    if rnd < rnd then
        Predator = α, y_Predator = y_α , key=1
    else
        Predator = β, y_Predator = y_β , key=2
    end
    if rnd < rnd then Prey = α else Prey = β end
    Prey := permuting(Prey)
    if rnd < rnd then R = 4 · rnd · (rnd − rnd) else R ~ Γ (4 · rnd, 1) end
    M_{1:N,1:D} = 1
    for q ← 1 to N · D do
        if rnd < (p · rnd) then M_{rndint(N),rndint(D)} = 0 end
    end
    if rnd < (p · rnd) then
        for i ← 1 to N do
            for j ← 1 to D do
                if rnd < (p · rnd) then
                    M_{i,j} = 1
                else
                    M_{i,j} = 0
                end
            end
        end
    end
    for i ← 1 to N do
        if ∑ M_i = D then M_{i,rndint(D)} = 0 end
    end
    // Mutation
    x = Predator + R · (Prey − Predator)
    for i ← 1 to N do
        for j ← 1 to D do
            // Crossover
            if M_{i,j} > 0 then x_{i,j} := Predator_{i,j} end
            // Boundary Control
            if (x_{i,j} < low_j) ∨ (x_{i,j} > up_j) then
                x_{i,j} := rnd · (up_j − low_j) + low_j
            end
        end
    end
    // Selection (Update)
    for i ← 1 to N do
        if f(x_i) < y_{i;Predator} then Predator_i := x_i, y_{i;Predator} := f(x_i) end
    end
    if key=1 then
        α := Predator, y_α := y_Predator
    else
        β := Predator, y_β := y_Predator
    end
    y_best = min(y_Predator) | best ∈ {1, 2, 3, ..., N}
    if y_best < GlobalMinimum then
        GlobalMinimum := y_best
        GlobalMinimizer := Predator_best
    end
end
```

Figure 3.8 Pseudo code of the ACS algorithm. (Civicioglu, 2013)

Some of the differences between the ACS algorithm and swarm intelligence algorithms as pointed by Civicioglu (2013) are given below:

- Differently from the algorithms such as Artificial Bee Colony (ABC), Particle Swarm Optimization (PSO) and Differential Search Algorithm (DSA), only two populations are used in the ACS algorithm.
- The trial pattern generation strategy of the ACS algorithm is different from other algorithms such as ABC, DSA and PSO.
- Only one pattern different from the target pattern is enough to generate the trial pattern in the ACS algorithm, while, in the Differential Evolution (DE) algorithm at least 3 patterns different from the target pattern are needed.
- The mutation crossover strategy of the ACS algorithm is different from other algorithms, such as DE.
- The boundary control mechanism of the ACS is different from the boundary control mechanism of the other algorithms.

3.7 Quadratic Approximation Operator

QA is an operator that finds the minima of a quadratic hyperspace, which is passing through three points in a D-dimensional space (Deep & Bansal, 2009). QA is a local search method. Mohan and Shanker (1994) showed that Random Search Technique (RST) that uses QA operator gives faster convergence rates. However, once it is trapped into local minima, it cannot get out of it easily. To implement the algorithm, first, the particle R_1 is chosen with the best objective function value. Then the particles R_2 and R_3 are randomly chosen from the remaining population. In the algorithm, R_1 is named as the leader individual, and the particle with the worst value is named as the active individual. Out of these three particles, at least two of them must be distinct. As a rule, the new point is accepted if it is better than the active individual and it is replaced with the active individual. Finally, the new minima point R^* of the quadratic surface passing through R_1, R_2 and R_3 can be calculated with the following formula,

$$R^* = 0.5\left(\frac{(R_2^2 - R_3^2) f(R_1) + (R_3^2 - R_1^2) f(R_2) + (R_1^2 - R_2^2) f(R_3)}{(R_2 - R_3) f(R_1) + (R_3 - R_1) f(R_2) + (R_1 - R_2) f(R_3)} \right) \tag{3.17}$$

where $f(R_1)$, $f(R_2)$ and $f(R_3)$ are the objection values at R_1, R_2 and R_3. In this thesis, QA operator is used along with ACS algorithm to solve the multi-objective EELD problem.

CHAPTER FOUR
THE PROPOSED ACS-QA ALGORITHM

In this chapter, we construct ACS-QA algorithm and implement it to solve the multi-objective EELD optimization problem. It contains valve-point effects, which make the problem non-linear, non-convex and non-continuous. Also, minimizing two objective functions at the same time makes the problem more involved and requires excess computational power.

ACS algorithm is well-tuned for solving a wide class of optimization problems thanks to its ability to explore large solution spaces and search in a family of candidate solution areas rather than just a single point. However, ACS algorithm is less adapted to fine-tuning solutions. Therefore, hybridizing the ACS algorithm with a local search algorithm is an appropriate approach to enhancing accuracy. In the present thesis, QA operator is coupled with the ACS algorithm to overcome the fine-tuning deficiency of the ACS algorithm. QA operator improves the solution because the algorithm satisfies the constraints with higher precision and searches all candidate solution areas to find the constrained optimum. Also, considering that the QA operator does not require extra functional evolution, it relieves the computational burden associated with hybrid evolutionary algorithms (EA), and hence, increases the rate of convergence (Wanner, Guimaraes, Takashi & Fleming, 2007).

In the proposed algorithm, first, ACS algorithm is executed. Then, during the execution of the algorithm, after finding the promising areas, the most promising area is selected as the best solution. Finally, QA operator comes in and the best solution of the ACS algorithm is used for the best particle with fitness value (R_1) in the QA operator. Result of the QA operator (R^*) is the ultimate solution that the proposed algorithm finds in an iteration. By doing this, QA operator enables us to explore more deeply to the candidate solution areas. Following steps constitute the proposed algorithm ACS-QA. Also, Flowchart of the proposed algorithm is given in Figure 4.1.

Step 1: Set upper and lower bounds, define cost coefficients, transmission loss coefficients and valve point coefficients for each generation. Determine population size, maximum number of generations and termination criterion. Set the penalty coefficient for the problem.

Step 2: Set the iteration counter to 1. Initialize α and β superorganisms and calculate the fitness values of the α and β superorganisms by taking into account the valve point effects, transmission losses and constraints depending on the objective function as given in the algorithmic form below:

for i=1 to N
  for j=1 to D
    $\alpha_{i,j:0} = r_1.(u_j - l_j) + l_j,\ \beta_{i,j:0} = r_1.(u_j - l_j) + l_j$
  end
  $y_{i;\alpha} = f(\alpha_i), y_{i;\beta} = f(\beta_i)$
end

Step 3: Determine the predator individuals and their respective fitness values as given in the algorithmic form below where r_1 and r_2 are random numbers selected from a gaussian distrubition between [0,1].

if $r_1 < r_2$
  $P_{red} = \alpha,\ \ y_{P_{red}} = y_a,\ \ key = 1$
else
  $P_{red} = \beta,\ \ y_{P_{red}} = y_\beta,\ \ key = 2$
end

Step 4: Determine the prey individuals and their respective fitness values as given in the algorithmic form below where *permute()* function randomly changes the places of row elements of prey individuals.

$$if\ r_1 < r_2\ \ P_{rey} = \alpha\ \ else\ \ P_{rey} = \beta\ \ end$$

$$P_{rey} = permute\ (P_{rey})$$

Step 5: Calculate scale factor (R) that controls speed of biological interaction as given in the algorithmic form below where $\Gamma(\text{a,b})$ is the gamma distribution with a shape parameter of $4\ .\ r_1$ and a scale parameter of 1.0.

$$\begin{aligned}
&if\ r_1 < r_2\ \ then \\
&\quad R = 4.r_1.(r_1 - r_2) \\
&else \\
&\quad R = \Gamma(4.r_1, 1) \\
&end
\end{aligned}$$

Step 6: Determine the passive individuals by applying binary valued integer map (M) as given in the algorithmic form below where *rndint()* function generates random integers between selected interval by using gauss distribution and p represents probability of biological interaction.

$$\begin{aligned}
&M = 1 \\
&for\ all\ elements\ in\ M \\
&\qquad if\ \ r_1 < (p.r_2)\ \ then\ \ M_{rndint(N)\ ,\ rndint(D)} = 0\ \ \ end \\
&end \\
&if\ \ r_1 < (p.r_2)\ \ then \\
&\quad for\ all\ elements\ in\ M \\
&\qquad\qquad if\ \ r_1 < (p.r_2)\ then\ \ M_{i,j} = 1\ \ else\ \ M_{i,j} = 0\ \ end \\
&\qquad end \\
&\quad end \\
&end \\
&for\ i{=}1\ to\ N \\
&\qquad if\ \ \sum_{j=1}^{D} M_i = D\ \ then\ \ M_{i,rndint(D)} = 0\ \ end \\
&end
\end{aligned}$$

Step 7: Compute the biological interaction locations (x) by using Eq. (3.16).

Step 8: From the solution matrix (y), select the row vector with the minimum fitness value (G_{best}). Determine the minima point R^* in Eq. (3.17) by selecting G_{best} as R_1 and two randomly chosen solution vectors from the remaning population as R_2 and R_3. If new solution vector is better than inferior solutions, update the perturbed solution vector and increment the generation counter by one.

Step 9: Update the biological interaction locations as given in the algorithmic form below:

for i=1 to N
for j=1 to D
if $M_{i,j} > 0$ *then* $x_{i,j} = P_{red_{i,j}}$ *end*
end
end

Step 10: Determine next generation of superorganisms as given in the algorithmic form below with the utilization of *key* parameter decided in Step 3.

if key=1 then
$\alpha = P_{red}, y_\alpha = y_{P_{red}}$
else
$\beta = P_{red}, y_\beta = y_{P_{red}}$
end

Step 11: Obtain the best fitness value of predator sub-superorganism. Register the fitness value and candidate solutions for the next generations.

Step 12: Repeat Step 3 to 11 until maximum generation number is met.

The proposed algorithm solves the optimization problem by applying the steps described above. After each run of the algorithm, following steps are considered to form the pareto-curve and solve the EELD problem:

Step 1: To solve the EELD problem, first, ELD and EED problems must be solved independently to calculate $C_F^{\max}$, $C_F^{\min}$, $C_E^{\max}$ and $C_E^{\min}$ in Eq. (2.17) by setting w to 1 and 0 respectively.

Step 2: Initialize w as 0.

Step 3: Solve Eq. (2.17) with the calculated minimum and maximum values in Step 1 and retain the solution.

Step 4: Repeat Step 3 by incrementing w by 0.05 each time until the value of w hits 1.

Step 5: Plot the pareto-curve and select the corresponding solution with the value of $w = 0.5$ as the best compromising solution.

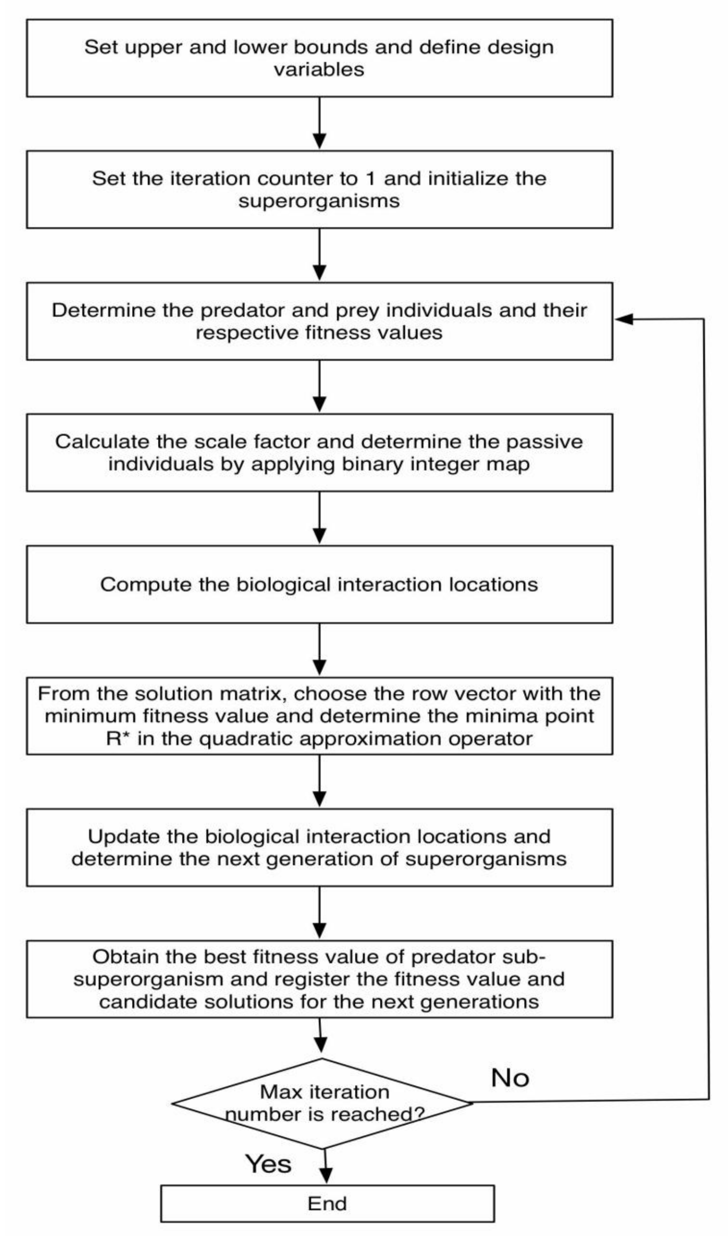


Figure 4.1 Flowchart of the proposed algorithm

CHAPTER FIVE
NUMERICAL ANALYSIS AND SIMULATION RESULTS

To test the effectiveness of the proposed algorithm, the proposed algorithm is applied to various unconstrained benchmark functions. Then, we analyze four different multi-objective EELD problems with 6, 10, 11 and 40 generating units. The proposed algorithm ACS-QA is implemented in Java language and executed on a 2.8 GHz Intel Dual Core personal computer with 4 GB RAM. Population size and iteration number are chosen as 10 and 9000 respectively. Since QA operator provides faster convergence rates, iteration number can be chosen at a lower value than usual.

5.1 Case Study 1

In this study, the proposed algorithm is compared with the known algorithms ACS (Civicioglu, 2013), QPSO (Jiang et al., 2014; Sun et al., 2004; Sun et al., 2005), Quantum behaved Particle Swarm Optimization algorithm, which is motivated by consepts from quantum mechanics and particle swarm optimization, ITHS (Kumar, Panda & Chang, 2012), Intelligent Tuned Harmony Search algorithm, an improvation of the Harmony Search (HS) algorithm which is based on a musician trying to find the perfect-tune, BBBC (Erol & Eksin, 2006), Big Bang Big Crunch algorithm which is inspired from the Big Bang and Big Crunch phenomenas happens in space, DS (Civicioglu, 2012), Differential Search algorithm which is inspired by migration of superorganisms utilizing the concept of stable-motion, BAT (Yang, 2010), Bat algorithm which is based on migration of a swarm of bats. All algorithms are implemented on six different unconstrained benchmark functions (namely; Sphere, Ackley, Griewank, Step, Schwefel and Alpine) to determine their solution accuracies and convergence speeds. Mathematical represantations and plots of these functions are given in Appendix A. Table 1 shows the results of all algorithms, comparatively (Here, 40 results are taken from each function solution and gaussian distributions are formed for each of those 40 results. Mean deviations and standard deviations are indicated in Table 5.1). It is clear that the proposed algorithm ACS-QA outperforms all the others. We emphasize that the hybrid ACS-QA performs better than the ACS itself. All this means that, the proposed algorithm is able to find correct solutions in the vicinity of the optimal value. Having shown effectiveness of ACS-QA in finding the solution, we now illustrate the convergence lines for different problems to reveal relative efficiency of ACS-QA in terms of number of function evaluations. For this, we solve Griewank, Ackley and Sphere functions by using ACS and ACS-QA and depict their convergence lines in Figure 5.1, Figure 5.2 and Figure 5.3, respectively. It is clearly seen that QA operator enhances the convergence rate of the ACS algorithm.

Table 5.1 Statistical properties of the solutions of unconstrained benchmark optimization problems

	Mean dev. + Std. dev.	Mean dev. + Std. dev.	Mean dev. + Std. dev.
	Sphere	Ackley	Griewank
QPSO	3.51E+01±1.28E+01	1.39E+01±1.46E+00	1.01E-00±5.01E-02
ITHS	5.26E-01±4.77E-01	1.59E+00±1.04E+00	5.83E-02±7.12E-02
BBBC	1.82E+01±1.05E+01	1.06E+01±2.75E+00	6.93E-01±2.43E-01
DS	3.63E+01±9.01E+00	1.33E+01±7.35E-01	1.02E+00±1.24E-02
BAT	2.52E-05±4.55E-06	5.90E+00±4.76E+00	8.87E-03±6.33E-03
ACS	2.05E-04±1.13E-04	4.25E-02±8.97E-03	5.94E-05±4.12E-05
ACS-QA	1.17E-09±2.13E-09	1.17E-04±6.17E-05	5.12E-09±7.01E-09
	Schwefel 2.22	Alpine	Step
QPSO	6.26E+01±1.59E+01	1.43E+01±3.37E+00	4.57E+01±1.22E+01
ITHS	6.79E+00±3.45E+00	1.25E+00±1.27E+00	5.33E+00±6.10E+00
BB-BC	6.17E+04±9.51E+05	2.42E+01±6.57E+00	1.93E+01±9.59E+00
DS	6.63E+01±6.83E+00	3.64E+01±3.01E+00	3.59E+01±6.32E+00
BAT	3.85E+04±1.12E+05	2.51E+01±1.19E+01	4.90E-05±7.94E-06
ACS	8.08E-02±1.80E-02	8.55E-01±4.02E-01	1.99E-04±9.03E-05
ACS-QA	2.14E-04±3.38E-04	1.10E-03±1.31E-03	2.49E-10±6.63E-10

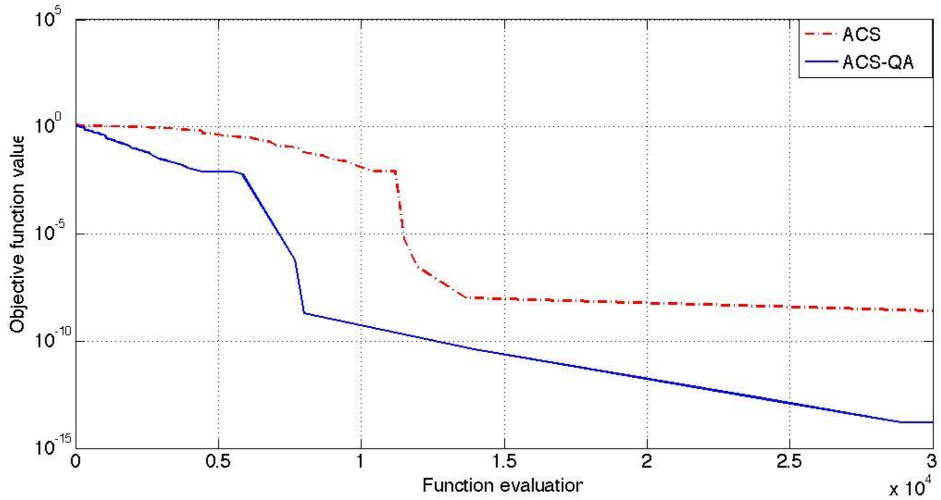


Figure 5.1 Convergence rates of the original ACS and the proposed ACS-QA algorithms for the Griewank function

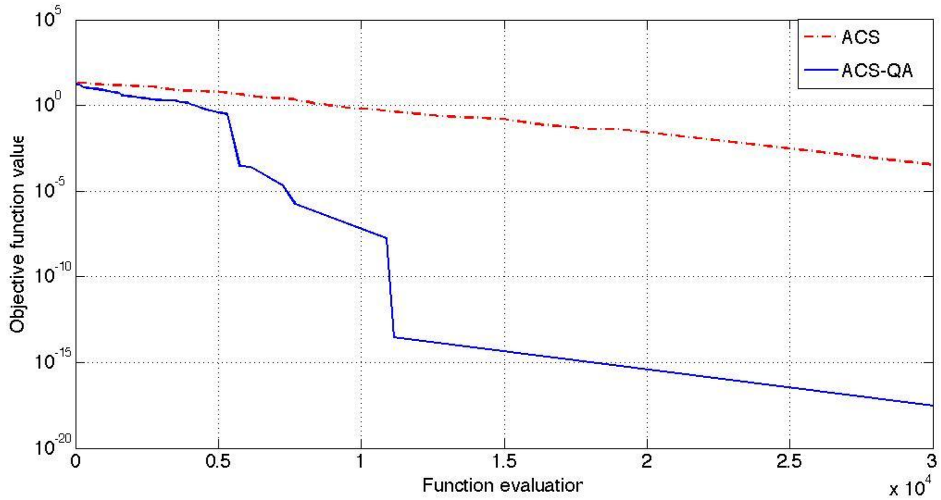


Figure 5.2 Convergence rates of the original ACS and the proposed ACS-QA algorithms for the Ackley function

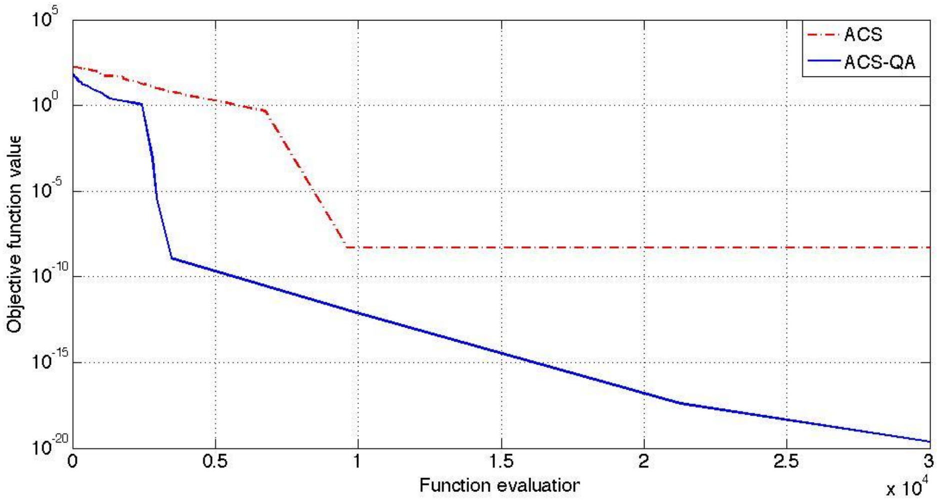


Figure 5.3 Convergence rates of the original ACS and the proposed ACS-QA algorithms for the Sphere function

5.2 Case Study 2

In this study, the proposed algorithm is implemented on a well-known network IEEE 30 bus including six thermal generating units to test the effectiveness of the algorithm. The system consists of six thermal units, 26 buses and 46 transmission lines (Farag, Al-Baiyat & Cheng, 1995). The load of the system was set to 2.834 p.u. on a 100 MVA base and both transmission losses, valve point effects and equality and inequality constraints are taken into account while solving the multi-objective EELD problem. Example diagram of the system is given in Figure 5.4. Pareto curves of the ACS and the proposed algorithm ACS-QA are given in Figure 5.5. It is seen that pareto curve of the proposed algorithm stays closer to the axes, which means that the proposed algorithm gives better results and outperforms the original ACS algorithm. The best compromised solutions and the corresponding power generations for each unit for the solution of the EELD problem by the proposed algorithm ACS-QA are given in Table 5.2 together with ACS, NPGA (Abido, 2003b), Niched Pareto Genetic Algorithm which is a variant of Genetic Algorithm for multiobjective optimization problems, PSO (Hemamalini & Simon, 2008), BBO (Roy & Hazra, 2014), MODE (Wu, Wang, Yuan & Zhou, 2010), Multiobjective Differential Evolution Algorithm which is a variation of Differential Algorithm designed for multiobjective optimization problems, IHBMO (Ghasemi, 2013), Interactive Honey

Bee Mating Optimization (IHBMO) algorithm, an improvement of the Honey Bee Mating algorithm which is based on mating of honey bees in a swarm, CIABC (Shayeghi & Ghasemi, 2014), Chaotic Improved Artificial Bee Colony algorithm which is an improvement of ABC algorithm based on chaotic search for candidate food position. It is seen in Table 5.2 that, the proposed algorithm finds better solutions than all except CIABC, in terms of the fuel cost. However, it is also seen that, in terms of the emission level, the proposed algorithm outperforms CIABC. In Table 5.3, we show the results in terms of power generation in units, fuel costs and emission levels for both ELD and EED problems by comparatively tabulating the proposed algorithm ACS-QA, ACS and HS (Sivasubramani & Swarup, 2011). It is seen that the proposed algorithm finds better results than the other algorithms in the ELD and EED problems.

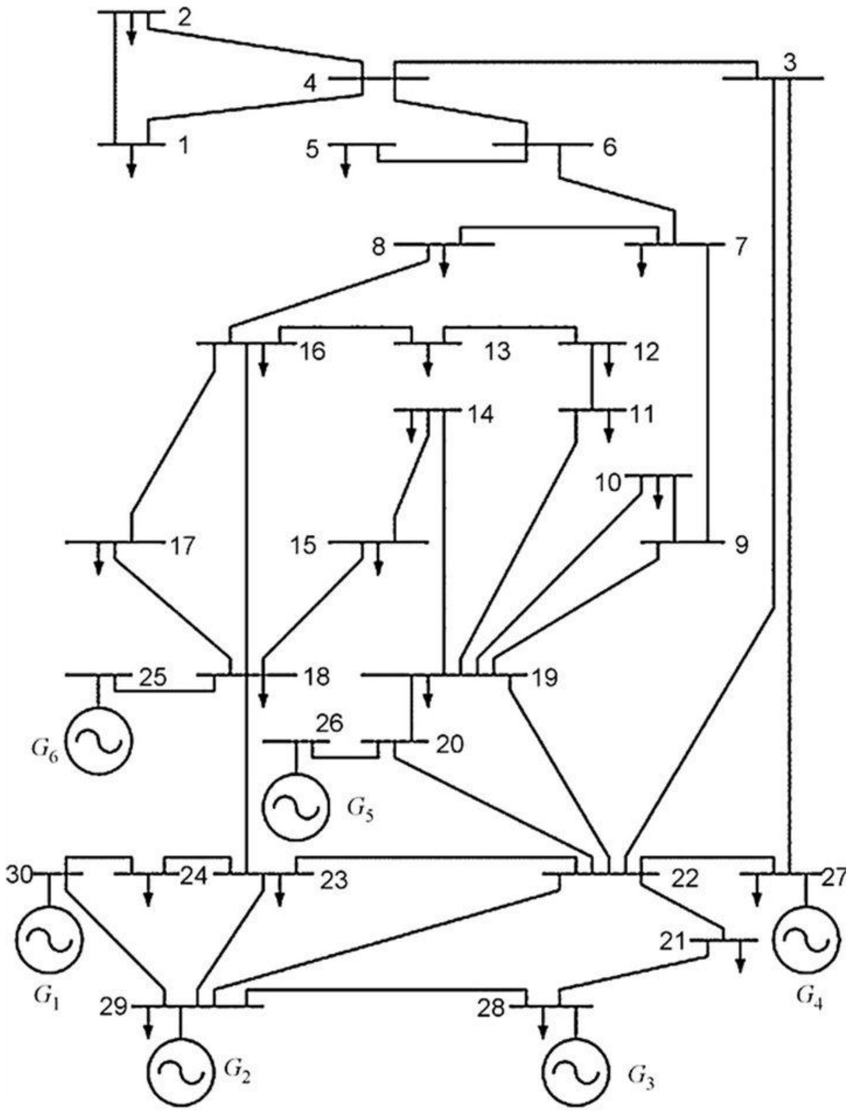


Figure 5.4 Single Line Diagram of IEEE-30 bus 6 generators system (El Sawy, Hendawy, El-Shorbagy,2013)

Table 5.2 Best Compromised Solutions of 6-unit system

Units	ACS-QA	ACS	NPGA	PSO	BBO	MODE	IHBMO	CIABC [4]
P_1	0.2467	0.2381	0.2227	0.1761	0.2625	0.2120	0.1874	0.2573
P_2	0.3670	0.3615	0.3787	0.2819	0.3770	0.3065	0.2825	0.2832
P_3	0.5622	0.5623	0.5560	0.5408	0.5760	0.6887	0.6876	0.6881
P_4	0.6938	0.7005	0.7147	0.7696	0.6735	0.6793	0.6367	0.6325
P_5	0.5453	0.5631	0.5500	0.6502	0.5377	0.5821	0.6813	0.6011
P_6	0.4264	0.4339	0.4424	0.4457	0.4270	0.3869	0.3605	0.3617
Fuel Cost	611.000	614.571	615.097	612.35	615.221	614.170	611.888	610.223
Emission	0.2012	0.2015	0.2020	0.2074	0.2002	0.2043	0.2057	0.2043

Table 5.3 Solutions of single-objective 6-units Economic Dispatch and Emission Dispatch problems

Units	Economic Dispatch			Emission Dispatch		
	ACS-QA	ACS	HS	ACS-QA	ACS	HS
P_1	0.1209	0.1221	0.0679	0.4107	0.4191	0.4397
P_2	0.2862	0.2834	0.3515	0.4636	0.4633	0.3908
P_3	0.5833	0.5891	0.5174	0.5444	0.5408	0.5506
P_4	0.9927	0.9902	0.8839	0.3902	0.3886	0.3774
P_5	0.5238	0.5204	0.5991	0.5448	0.5479	0.5420
P_6	0.3518	0.3538	0.4317	0.5154	0.5098	0.5021
Fuel Cost	605.866	606.002	606.2858	646.205	646.759	647.434
Emission	0.2207	0.2205	0.2148	0.1941	0.1941	0.1951

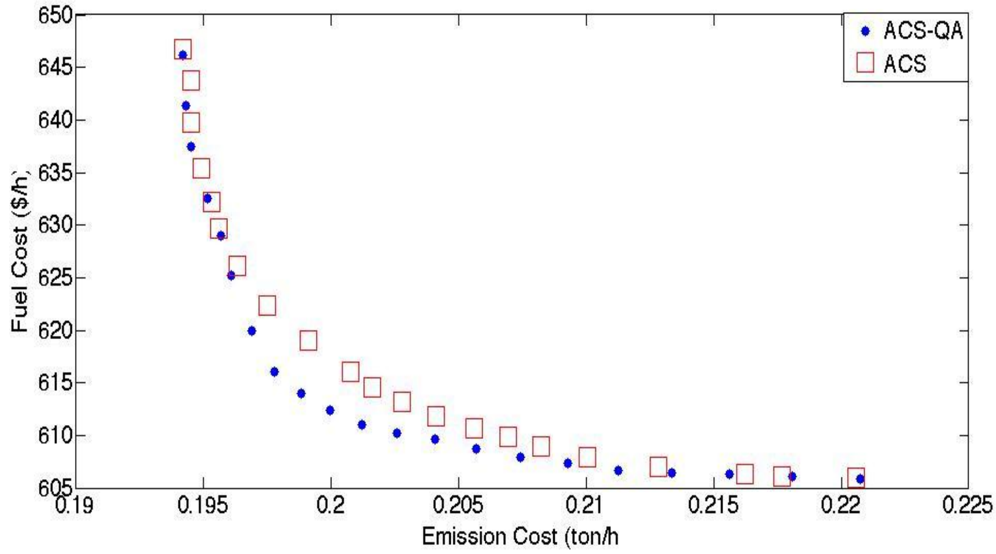


Figure 5.5 Pareto curves of the original ACS and proposed ACS-QA algorithms for the 6 generation units system

5.3 Case Study 3

A 10 generators system with valve point effects both on fuel cost and emission objective functions is considered. Transmission losses, equality and inequality are also taken into consideration for this study. Total load demand is 2000 MW. Pareto curves obtained by solving the multi objective EELD problem with the proposed algorithm ACS-QA and ACS algorithms are given in Figure 5.6. It can be seen that pareto curve of ACS-QA is closer to the axes. Therefore, it can be concluded that ACS algorithm gives better results when hybridized with QA operator. Best compromising solutions and corresponding electric generation in units obtained when the EELD problem is solved with the proposed algorithm, ACS, MODE (Basu, 2011), GSA (Guvenc, Sonmez, Duman & Yorukeren, 2012), Gravitational Search Algorithm which is based on law of gravity and mass interactions, RCCRO (Bhattacharjee et al., 2014), Real Coded Chemical Reaction Algorithm which takes inspiration from energy states of the molecules during a chemical reaction, PDE (Basu, 2011), Pareto based Differential Evoution algorithm which is an improved version of Differential Evolution algorithm for multiobjective optimization problems, NSGA-II (Basu, 2011), Non-dominated Sorting Genetic Algorithm which is an improvement of Genetic Algorithms with non-dominated sorting approach for multiobjective optimization problems, MHSA (Jeddi & Vahidinasab, 2014), Mosquito host-seeking algorithm which is inspired from host-seeking behavior of mosquitos, are given in Table 5.4. Once again, it is seen that the proposed algorithm ACS-QA gives better results in terms of the fuel cost yet worse results in terms of the emission level. Also, individual solutions of single-objective ELD and EED problems with the proposed algorithm ACS-QA, ACS and RCCRO (Bhattacharjee et al., 2014) are given in Table 5.5. The proposed algorithm performs better than the other algorithms for both ELD and EED problems.

Table 5.4 Best compromised solutions of 10-unit system

Units	ACS-QA	ACS	MODE	GSA	RCCRO	PDE	NSGA-II	MHSA
P_1	54.9998	54.7798	54.9487	54.9992	55.0000	54.9853	51.9515	54.4132
P_2	79.9998	79.7031	74.5821	79.9586	80.0000	79.3803	67.2584	70.6736
P_3	86.5405	90.7377	79.4294	79.4341	85.6453	83.9842	73.6879	97.0719
P_4	84.8731	84.1959	80.6875	85.0000	84.1259	86.5942	91.3554	86.4019
P_5	127.1193	123.0420	136.8551	142.1063	136.5034	144.4386	134.0522	138.0141
P_6	144.1107	168.1831	172.6393	166.5670	155.5801	165.7756	174.9504	162.4903
P_7	299.9942	292.0955	283.8233	292.8749	300.0000	283.2122	289.4350	283.6421
P_8	320.7936	315.6864	316.3407	313.2387	316.6746	312.7709	314.0556	311.5283
P_9	441.8326	436.3438	448.5923	441.1775	434.1252	440.1135	455.6978	439.0945
P_{10}	444.5285	439.4543	436.4287	428.6306	436.5724	432.6783	431.8054	440.7168
Fuel Cost	112,690.54	113,113.53	113,480.00	113,490.00	113,355.74	113,510.00	113,540.00	113,290.00
Emission	4203.8881	4158.7420	4124.9000	4111.4000	4121.0684	4111.4000	4130.2000	4153.3000

Table 5.5 Solutions of single-objective 10-units Economic Dispatch and Emission Dispatch problems

Units	Economic Dispatch			Emission Dispatch		
	ACS-QA	ACS	RCCRO	ACS-QA	ACS	RCCRO
P_1	54.9999	54.9786	55.0000	54.9999	54.9723	55.0000
P_2	79.9999	79.9809	79.9999	79.9999	79.6051	80.0000
P_3	106.9399	106.8501	106.9220	81.1341	81.0165	81.1342
P_4	100.5762	99.9306	100.5426	81.3637	81.7512	81.3637
P_5	81.5017	82.7102	81.5216	159.9999	159.9788	160.0000
P_6	83.0208	82.8818	83.0528	239.9999	239.9592	240.0000
P_7	299.9999	299.9293	299.9999	294.4851	293.5266	294.4851
P_8	339.9999	339.9363	339.9999	297.2701	298.5785	297.2701
P_9	469.9999	469.9088	469.9999	396.7658	396.3209	396.7657
P_{10}	469.9999	469.9252	469.9999	395.5761	395.8805	395.5763
Fuel Cost	111,497.630	111,499.510	111,497.6319	116,412.444	116,408.346	116,412.4441
Emission	4572.195	4567.810	4571.9552	3932.243	3932.485	3932.2433

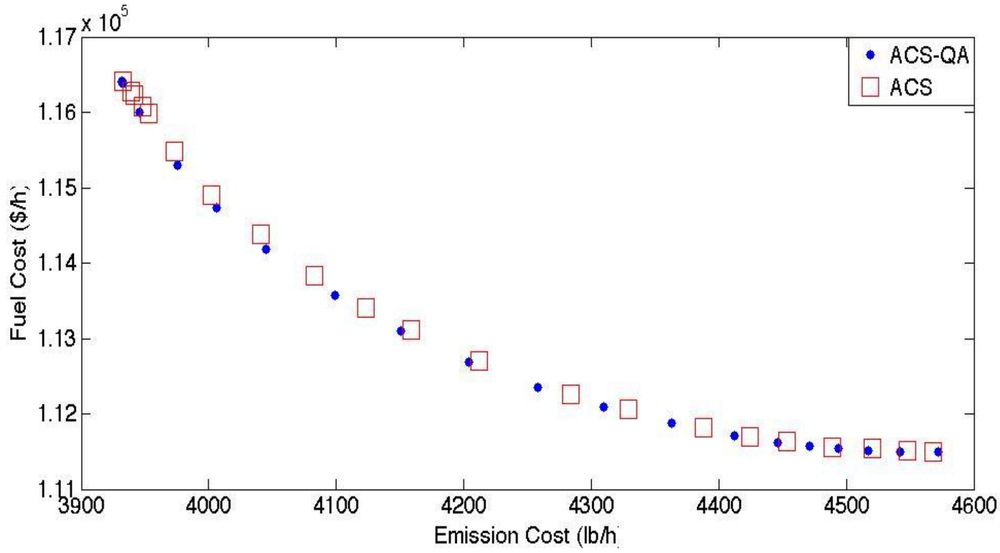


Figure 5.6 Pareto curves of the original ACS and proposed ACS-QA algorithms for the 10 generation units system

5.4 Case Study 4

In this study, a power system with 11-generation units is considered. Valve point effects are taken into account for both fuel cost and emission objective functions. Transmission losses are neglected in this case. Total load demand is 2500 MW. Pareto curves obtained by solving the multi objective problem with both the proposed ACS-QA and the original ACS algorithms are presented in Figure 5.7. It is seen that the proposed algorithm's pareto curve is closer to axes, thus outperforms the ACS algorithm. Also, in Table 5.6, best compromised results and corresponding power generations for each generator obtained by solving the multi-objective problem with the proposed algorithm ACS-QA, ACS and GABSC (Guvenc, 2010), Genetic Algorithm method Based on Similarity Crossover in which new generation is created by genetic similarity measurement between farther and mother, are presented. The proposed algorithm gives better results than the other algorithms in terms of fuel costs but is outperformed by GABSC algorithm in terms of the emission level.

Table 5.6 Best compromised solutions of 11-units system

Units	ACS-QA	ACS	GABSC
P_1	122.8613	122.6303	138.8618
P_2	103.2495	95.5445	112.1312
P_3	143.0472	156.3786	146.7169
P_4	212.7189	219.9551	222.1041
P_5	170.0694	165.3001	137.1962
P_6	209.0082	229.6278	217.3208
P_7	168.2317	151.3600	140.4711
P_8	369.3839	382.1478	348.9008
P_9	319.3833	324.0510	326.5188
P_{10}	357.4635	354.8310	363.5275
P_{11}	324.5825	298.1739	346.2508
Fuel Cost	12,389.604	12,394.501	12,423.770
Emission(ton)	2030.286	2038.393	2003.030

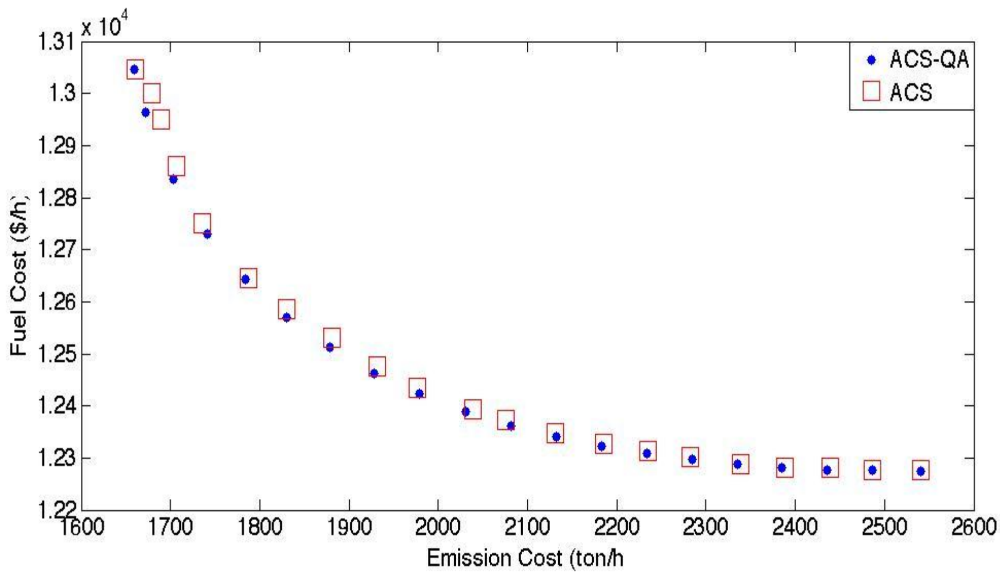


Figure 5.7 Pareto curves of the original ACS and proposed ACS-QA algorithms for the 11 generation units system

5.5 Case Study 5

This time, a 40-unit power system with valve point effects on both fuel cost and emission objective functions is considered. Transmission losses have been neglected and equality and inequality constraints are taken into account. Total load demand is 10,500 MW. Pareto curves by solving the multi-objective problem with the proposed ACS-QA and ACS algorithms are presented in Figure 5.8. It is seen that the proposed algorithm outperforms the ACS algorithm. Best compromised solutions and the corresponding power generations in units obtained by solving the multi-objective problem with the proposed algorithm ACS-QA, ACS, RCCRO (Bhattacharjee et al., 2014), QOTLBO (Roy & Bhui, 2013), which is an improvement of the Teaching Learning Based Optimization algorithm with quasi-Oppositional Based Learning concepts, GSA (Roy & Bhui, 2013) are tabulated in Table 5.7. The proposed algorithm finds worse results than RCCRO and QOTLBO algorithms in terms of fuel cost, however, outperforms them in terms of the emission level. Individual solutions of the ELD and EED problems with the proposed ACS-QA, ACS and RCCRO (Bhattacharjee et al., 2014) algorithms are presented in Table 5.8. It is seen that the proposed algorithm ACS-QA gives better results than the other algorithms for both ELD and EED problems. Also, solutions of single-objective EED and ELD problems with the proposed ACS-QA, ACS, DE (Sharma, Samantaray, Mohanty & Rout, 2011), Differential Evolution algorithm which is based on maintaining a population of candidate solutions and combining them with the new solutions according to some rules, MBFA (Hota et al., 2010), Modified Bacterial Foraging Algorithm, which is an improvement of Bacterial Foraging algorithm by adding a dynamic chemotactic step, MODE (Sharma et al., 2011), NSGA-II (Sharma et al., 2011), IABC (Aydin et al., 2011), IABC-LS (Aydin et al., 2011), ABCDP (Aydin et al., 2011) and ABCDP-LS (Aydin et al., 2011) algorithms are given in Table 5.9. The proposed algorithm ACS-QA is seen to give better results than the other algorithms.

Table 5.7 Best compromised solutions of 40-units system

Units	ACS-QA	ACS	RCCRO	QOTLBO	GSA
P_1	113.9890	111.3116	111.0511	114.0000	113.9989
P_2	113.9831	112.1779	111.1804	114.0000	113.9896
P_3	119.9962	102.5186	97.4009	120.0000	119.9995
P_4	179.7336	179.4192	179.7329	179.7593	179.7857
P_5	96.9947	93.5415	96.9995	97.0000	97.0000
P_6	139.8771	135.5289	139.9999	140.0000	139.0128
P_7	299.9934	283.8897	259.6001	300.0000	299.9885
P_8	285.9143	284.7419	284.5999	298.9093	300.0000
P_9	295.2811	285.0565	284.5997	300.0000	296.2025
P_{10}	130.0212	181.1877	130.0003	130.0996	130.3850
P_{11}	318.3612	245.0746	243.6001	243.7055	245.4775
P_{12}	318.3540	305.0227	243.5997	318.4741	318.2101
P_{13}	394.2933	394.0219	394.2794	394.4004	394.6257
P_{14}	394.2950	395.7532	394.2797	394.3418	395.2016
P_{15}	394.2796	393.2459	394.2793	394.2703	306.0014
P_{16}	394.2875	396.0321	394.2797	394.4013	395.1005
P_{17}	489.1414	474.2753	489.2794	489.3143	489.2569
P_{18}	489.2636	479.3932	489.2792	489.3548	488.7598
P_{19}	421.5591	484.8716	511.2789	511.1648	499.2320
P_{20}	470.4998	489.6450	511.2794	421.8134	455.2821
P_{21}	433.5977	434.0667	433.5193	434.5654	433.4520
P_{22}	433.5595	437.2441	433.5199	434.5536	433.8125
P_{23}	433.7296	442.8724	433.5210	433.9734	445.5136
P_{24}	433.5698	437.5403	433.5199	433.7659	452.0547
P_{25}	433.5478	436.0871	433.5205	434.9881	492.8864
P_{26}	433.5697	434.9823	533.5205	434.1780	433.3695
P_{27}	10.1593	10.6135	10.0000	10.0574	10.0026
P_{28}	10.1786	15.1337	10.0000	10.3295	10.0246
P_{29}	10.1308	12.8350	10.0001	10.0147	10.0125
P_{30}	96.9962	95.9513	96.9999	97.0000	96.9125
P_{31}	189.9500	187.1076	189.9999	190.0000	189.9689
P_{32}	189.6785	173.1965	189.9999	190.0000	175.0000
P_{33}	189.7372	170.2031	189.9999	190.0000	189.0181
P_{34}	199.9978	199.0226	199.9999	200.0000	200.0000
P_{35}	199.9968	199.7197	199.9998	200.0000	200.0000
P_{36}	199.9972	199.4629	199.9999	200.0000	199.9978
P_{37}	109.9861	101.0463	109.9999	110.0000	109.9969
P_{38}	109.9289	98.8350	109.9998	110.0000	109.0126
P_{39}	109.9922	107.4239	109.9999	110.0000	109.4560
P_{40}	421.5737	479.9567	511.2793	421.5651	421.9987
Fuel Cost	125,585.004	126,662.509	124,420.951	125,161.000	125,782.000
Emission	197,025.541	200,829.089	229,395.900	206,490.400	210,932.900

Table 5.8 Solutions of single-objective 40-units Economic Dispatch and Emission Dispatch problems

Units	Economic Dispatch			Emission Dispatch		
	ACS-QA	ACS	RCCRO	ACS-QA	ACS	RCCRO
P_1	110.8687	112.4920	110.7998	113.9999	113.9999	114.0000
P_2	111.0013	112.9250	110.7998	113.9999	113.9999	114.0000
P_3	97.3999	97.3999	97.3999	119.9999	113.9999	120.0000
P_4	179.7331	179.7331	179.7331	169.3679	169.4326	169.3680
P_5	92.4706	88.9759	87.7999	96.9999	96.9999	97.0000
P_6	139.9999	139.9999	140.0000	124.2574	124.2018	124.2574
P_7	259.5996	259.5996	259.5997	299.7114	299.5729	299.7114
P_8	284.5996	284.5996	284.5997	297.9148	297.8595	297.9149
P_9	284.5996	284.5996	284.5997	297.2601	297.2153	297.2601
P_{10}	130.0000	130.0000	130.0000	130.0000	130.0095	130.0000
P_{11}	168.7998	168.7998	94.0000	298.4101	298.4385	298.4101
P_{12}	168.7998	168.8000	94.0000	298.0259	297.9915	298.0260
P_{13}	214.7597	214.7597	214.7598	433.5576	433.4732	433.5576
P_{14}	394.2793	304.5195	394.2794	421.7284	421.7731	421.7284
P_{15}	394.2793	394.2606	394.2794	422.7796	422.7951	422.7796
P_{16}	304.5195	394.2793	394.2794	422.7796	423.0747	422.7796
P_{17}	489.2793	489.2793	489.2794	439.4128	439.4660	439.4129
P_{18}	489.2793	489.2793	489.2794	439.4028	439.1308	439.4029
P_{19}	511.2793	511.2793	511.2794	439.4128	439.4964	439.4128
P_{20}	511.2793	511.2793	511.2794	439.4128	439.4674	439.4129
P_{21}	523.2793	523.2793	523.2794	439.4463	439.4694	439.4464
P_{22}	523.2793	523.2793	523.2794	439.4463	439.3803	439.4464
P_{23}	523.2793	523.2793	523.2794	439.7720	439.8533	439.7721
P_{24}	523.2793	523.2793	523.2794	439.7720	439.8838	439.7721
P_{25}	523.2793	523.2793	523.2794	440.1117	440.0880	440.1118
P_{26}	523.2793	523.2793	523.2794	440.1117	440.0495	440.1118
P_{27}	10.0000	10.0000	10.0000	28.9937	29.0103	28.9937
P_{28}	10.0000	10.0000	10.0000	28.9937	29.0380	28.9937
P_{29}	10.0000	10.0000	10.0000	28.9937	29.1187	28.9937
P_{30}	87.8169	92.0588	87.7999	96.9999	96.9999	97.0000
P_{31}	189.9999	189.9999	190.0000	172.3319	172.1331	172.3319
P_{32}	189.9999	189.9999	190.0000	172.3318	172.4129	172.3319
P_{33}	189.9999	189.9999	190.0000	172.3319	172.3404	172.3319
P_{34}	164.7998	164.7998	164.7998	199.9999	199.9999	200.0000
P_{35}	164.7998	164.7998	194.3978	199.9999	199.9999	200.0000
P_{36}	164.7998	164.7998	200.0000	199.9999	199.9999	200.0000
P_{37}	109.9999	109.9999	110.0000	100.8383	100.7965	100.8384
P_{38}	109.9999	105.7222	110.0000	100.8383	100.8868	100.8384
P_{39}	109.9999	109.9999	110.0000	100.8383	100.7583	100.8384
P_{40}	511.2793	511.2793	511.2794	439.4128	439.3774	439.4129
FuelCost	121,371.560	121,411.816	121,412.535	129,954.643	129,961.797	129,995.2
Emission	356,430.784	355,812.992	359,901.381	176,672.264	176,682.697	176,682.2

Table 5.9 Solutions of single-objective 40-units Economic Dispatch and Emission Dispatch problems

Units	Economic Dispatch		Emission Dispatch	
	Cost	Emission	Cost	Emission
ACS-QA	121,371.560	356,430.784	129,954.643	176,672.264
ACS	121,411.816	355,812.992	129,961.797	176,682.697
DE	121,840.000	374,790.000	129,960.000	176,680.000
MBFA	121,415.650	356,424.490	129,995.000	176,682.260
MODE	121,836.980	374,790.560	129,956.090	176,683.270
NSGA-II	124,963.500	262,489.270	129,965.890	176,691.960
IABC	121,414.800	356,421.700	129,995.470	176,682.250
IABC-LS	121,412.720	359,900.830	129,995.200	176,682.250
ABCDP	121,412.820	359,900.700	129,995.410	176,682.250
ABCDP-LS	121,412.740	359,901.170	129,995.490	176,682.250

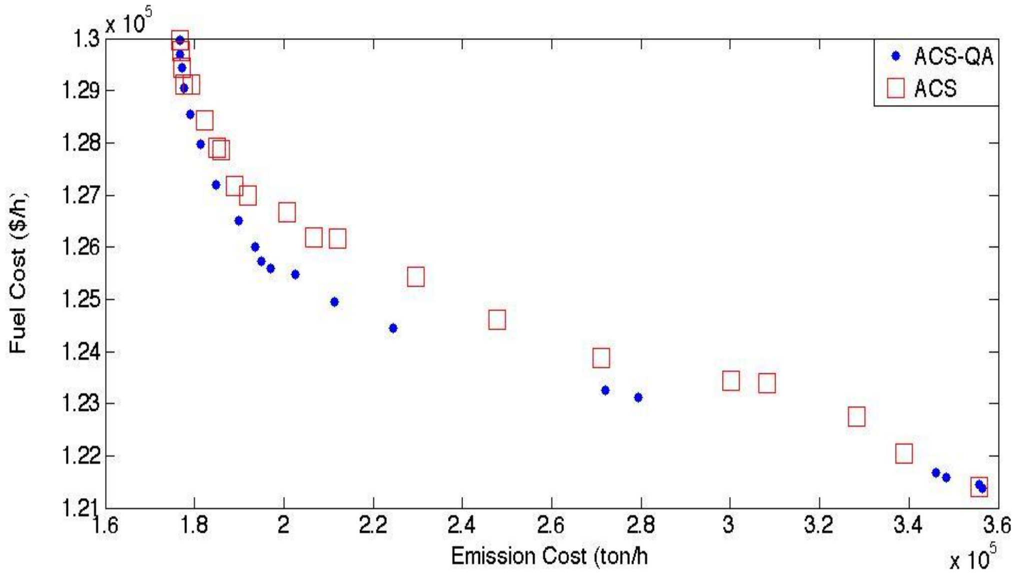


Figure 5.8 Pareto curves of the original ACS and proposed ACS-QA algorithms for the 40 generation units system

CHAPTER SIX
CONCLUSION

Metaheuristic optimization algorithms are powerful tools to deal with the optimization problems. Most of them are inspired from the nature and relies on stochastic processes such as randomization. They do not guarantee to find the globally optimum solutions all the time, however, with efficient metaheuristic algorithms like ACS, there is a greater chance to find the global solution. This thesis showed that ACS is an efficient optimization algorithm for solving global optimization problems.

In this thesis, ACS algorithm is hybridized with QA operator to construct an efficient and effective algorithm. By hybridizing two or more algorithms, increases the accuracy and efficiency of the algorithm. Which means the quality of solution increaes while the function evaluation count, which is needed to find the solution decreases. Hybridized ACS-QA algorithm used in this thesis once again proved that hybridization is an effective method for increasing the effectivity of the metaheuristic algorithms. We have shown that the algorithm, ACS-QA, outperforms the original ACS and several other optimization techniques in application to the multi-objective EELD problem with practical constraints like valve-point effects, prohibited operating zones and transmission losses, which are typical of the real-world power systems. We have revealed this by a variety of numerical examples in Chapter 5. The ACS is a known global search algorithm for solving optimization problems, however, it may not be effective for solving multi objective problems like EELD, as we have shown here. It is with hybridization of ACS with QA, the algorithm ACS-QA, that we obtain an efficient algorithm, which is successfully, implemented in four standard power systems and some benchmark optimization functions. The results show that ACS-QA is a powerful algorithm for solving multi objective optimization problems.

Also, WSM method, which is a tool for handling the multiobjectivity in the optimization problems, is introduced. In the WSM method, each objective function is evaluated with a weight and tradeoff between the objectives are defined with that weights. According to the WSM, sum of the all weights must be one. And the best comprimising solution is often where all weights are equal to each other. This thesis proved that WSM is a very efficient tool for handling multiobjectivity of the optimization problems.

As a future work, different hybrid optimization algorithms or the way of handling the multiobjectivity can be changed to find more desirable solutions for the multiobjective EELD problem. Also, some components may be included to the EELD system such as wind farms or fuel cells, which will change the cost function of the problem.

REFERENCES

Abido, M. A. (2003). A novel multiobjective evolutionary algorithm for environmental economic power dispatch. *Electric Power Systems Research, 65*(1), 71-81.

Abido, M. A. (2003). A niched Pareto genetic algorithm for multiobjective environmental economic power dispatch. *International Journal of Electrical Power Energy Systems*, *25*(2), 97-105.

Abido, M. A. (2003). Environmental/economic power dispatch using multiobjective evolutionary algorithms. *IEEE Transactions on Power Systems*, *18*(4), 1529-1537.

Abido, M. A. (2009). Multiobjective particle swarm optimization for environmental economic dispatch problem. *Electric Power System Research*, *79*(7), 1105-1113.

Aydin, D., Ozyon, S., Yasar, C., & Liao, T. (2014). Artificial bee colony algorithm with dynamic population size to combined economic and emission dispatch problem. *International Journal of Electrical Power and Energy Systems*, *54,* 144-153.

Basu, M. (2011). Economic environmental dispatch using multi-objective differential evolution. *Applied Soft Computing*, *11*(2), 2845-2853.

Bhattacharjee, K., Bhattacharya, A., & Halder nee Dey, S. (2014). Solutions of Economic Emission Load Dispatch problems of power systems by Real Coded Chemical Reaction algorithm. *International Journal of Electrical Power and Energy Systems*, *59*, 176-187.

Chase, N., Rademacher, M., Goodman, E., Averill, R., & Sidhu, R. (2009). A benchmark study of multi-objective optimization methods, *Red Cedar Technology Technical Notes Review 06.09*.

Civicioglu, P. (2012). Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm. *Computers and Geosciences*, *46*, 229-247.

Civicioglu, P. (2013). Artificial cooperative search algorithm for numerical optimization problems. *Information Sciences*, *229*, 58-76.

Civicioglu, P., & Besdok, E. (2013). A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms. *Artificial Intelligence Review*, *39*(4), 315-346.

Deep, K., & Bansal, J. C. (2009). Hybridization of particle swarm optimization with quadratic approximation. *OPSEARCH*, *46*(1), 3-24.

Dillon, T., Edwin, K., Kochs, H., & Taud, R. (1978). Integer programming commitment with probabilistic reserve determination. *IEEE Transactions on Power Apparatus and Systems*, *97*(6), 2154-2166.

Erol, O. K., & Eksin, I. (2006). A new optimization method: big bang-big crunch. *Advances in Engineering Software*, *37*(2), 106-111.

Farag, A., Al-Baiyat, S., & Cheng, T. C. (1995). Economic load dispatch multiobjective optimization procedures using linear programming techniques. *IEEE Transactions on Power Systems*, *10*, 731-738.

Fraga, E. S., Yang, L., & Papageorgiou, L. G. (2012). On the modelling of valve point loadings for power electiricty dispatch. *Applied Energy*, *91*(1), 301-303.

Gaing, Z-L. (2003). Particle swarm optimization to solving the economic dispatch with considering the generator constraints. *IEEE Transactions on Power Systems*, *18*(3), 1187-1195.

Ghasemi, A. (2013). A fuzzified multi objective interactive honey bee mating optimization for environmental/economic power dispatch with valve point effect. *International Journal of Electrical Power and Energy Systems*, *49*, 308-321.

Guvenc, U. (2010). Combined economic emission dispatch solution using genetic algorithm based on similarity crossover. *Scientific Research and Essays*, *5*(17), 2451-2456.

Guvenc, U., Sonmez, Y., Duman, S., & Yorukeren, N. (2012). Combined economic and emission dispatch solution using gravitational search algorithm. *Scientia Iranica*, *19*(6), 1754-1762.

Hemamalini, S., & Simon, S. P. (2008). Emission constrained economic dispatch with valve-point effect using particle swarm optimization. *Proceedings of the IEEE Technical Conference*, 1-6.

Hota, P. K., Barisal, A. K., & Chakrabarti, R. (2010). Economic emission load dispatch through fuzzy based bacterial foraging algorithm. *Electrical Power and Energy Systems*, *32*(7), 794-803.

Jadoun, V. K., Gupta, N., Niazi, K. R., & Swarnkar, A. (2014). Dynamically controlled particle swarm optimization for large-scale nonconvex economic dispatch problems. *International Transactions on Electrical Energy Systems*, DOI: 10.1002/etep.2022.

Jeddi, B., & Vahidinasab, V. (2014). A modified harmony search method for environmental/economic load dispatch of real-world power systems. *Energy Conversion and Management*, *78*, 661-675.

Jiang, S., Ji, Z., & Shen, Y. (2014). A novel hybrid particle swarm optimization and gravitational search algorithm for solving emission load dispatch problems with various practical constraints. *Electrical Power and Energy Systems*, *55*, 628-644.

Kaur, A. (2011). *Analysis and comparison of Economic Load Dispatch using Genetic Algorithm and Particle Swarm Optimization*. Master's Thesis, Thapar University, Thapar.

Khamsawang, S., & Jiriwibhakorn, S. (2010). DSPSO-TSA for economic dispatch problem with nonsmooth and noncontinuous cost functions. *Energy Conversion and Management*, *51*(2), 365-375.

Kumar, P. Y. R., Panda, S. K., & Chang, C. S. (2012). An intelligent tuned harmony search algorithm for optimization. *Information Sciences*, *196*, 47-72.

Lam, A. Y. S., Li, V. O. K., & Yu, J. J. Q. (2012). Real-coded chemical reaction optimization. *IEEE Transactions on Evolutionary Computing*, *16*(3), 339-353.

Lee, K., & Geem, Z. (2005). A new metaheuristic algorithm for continuous engineering optimization: harmony search theory and practice. *Computer Methods in Applied Mechanics and Engineering*, *194*(36), 3902-3933.

Mohan, C., & Shanker, K. (1994). A random search technique for global optimization based on quadratic approximation. *Asia Pacific Journal of Operations Research*, *11*, 93-101.

Multiobjective optimization. (n.d.). Retrieved May 20, 2015, from http://en.wikipedia.org/wiki/Multi-objective_optimization.

Ozyon, S., Temurtas, H., Durmus, B., & Kuvat, G. (2012). Charged system search algorithm for emission constrained economic power dispatch problem. *Energy*, *46*(1), 420-430.

Papageorgiou, L., & Fraga E. (2007). A mixed integer quadratic programming formulation for the economic dispatch of generators with prohibited operating zones. *Electric Power System Research*, *77*(10), 1292-1296.

Rajagopalan, A., Sengoden, V., & Govindasamy, R. (2014). Solving economic load dispatch problems using chaotic self-adaptive differential harmony search algorithm. *International Transactions on Electrical Energy Systems*, DOI: 10.1002/etep.1877.

Rajasomashekar, S., & Aravindhababu, P. (2012). Biogeography based optimization technique for best-compromise solution of economic emission dispatch. *Swarm and Evolutionary Computation*, *7*, 47-57.

Rajesh, K., Abhinav, S., Rudesh, K., & Panda, S. K. (2012). A novel multi-objective directed bee colony optimization algorithm for multi-objective emission constrained economic power dispatch. *International Journal of Electrical Power and Energy Systems*, *43*(1), 41-50.

Rajkumar, M. (2014). *Combined Economic Emission Dispatch using Modified Multi-Objective Genetic Algorithm*. Phd Thesis, Kalasalingam University, Kalasalingam.

Roy, P. K., & Bhui, S. (2013). Multi-objective quasi-oppositional teaching learning based optimization for economic emission load dispatch problem. *International Journal of Electrical Power and Energy Systems*, *53*, 937-948.

Roy, P. K, & Hazra, S. (2014). Economic emission dispatch for wind-fossil-fuel-based power system using chemical reaction optimisation. *International Transactions on Electrical Energy Systems,* DOI: 10.1002/etep.2033.

Sharma, R., Samantaray, P., Mohanty, D. P., & Rout, P. K. (2011). Environmental economic load dispatch using multiobjective differential evolution algorithm. *Proceeding of International Conference on Energy, Automation and Signal (ICEAS)*, 1-7.

Shayeghi, H., & Ghasemi, A. (2013). A modified artificial bee colony based on chaos theory for solving non-convex emission/economic dispatch. *Energy Conversion and Management*, *79*, 344-354.

Silva, Md. A. Ce., Klein, C. E., Mariani, V. C., & Coelho, Ld. S. (2013). Multiobjective scatter search approach with new combination scheme applied to solve environmental/economic dispatch problem. *Energy*, *53*, 14-21.

Simon, D. (2013). *Evolutionary optimization algorithms* (1st ed.). New Jersey: John Wiley and Sons.

Sivasubramani, S., & Swarup, K. S. (2010). Hybrid SOA-SQP algorithm for dynamic economic dispatch wtith valve-point effects. *Energy, 35*(12), 5031-5036.

Sivasubramani, S., & Swarup, K. S. (2011). Environmental/Economic dispatch using multi-objective harmony search algorithm. *Electric Power Systems Research*, *81*, 1778-1785.

Sivanagaraju, S., & Sreenivasan, G. (2009). *Power system operation and control* (1st ed.). India: Pearson.

Soumitra, M., Aniruddha, B., & Sunita, H. D. (2012). Multi-objective economic emission load dispatch solution using gravitational search algorithm and considering wind power penetration. *International Journal of Electrical Power and Energy Systems*, *44*(1), 282-292.

Sun, J., Feng, B., & Xu, W. B. (2004). Particle swarm optimization with particles having quantum behavior. *IEEE Congress on Evolutionary Computation*, *1*, 325-331.

Sun, J., Feng, B., & Xu, W. B. (2005). Adaptive parameter control for quantum behaved particle swarm optimization on individual level. *IEEE International Conference on Systems, Man and Cybernetics*, *4*, 3049-3054.

Venkatesh, P., Gnanadass, R., & Padhy, N. P. (2003). Comparison application of evolutionary programming techniques to combined economic emission dispatch with line flow constraints. *IEEE Transactions of Power Systems*, *18*(2), 688-697.

Wanner, E. F., Guimaraes, F. G., Takashi, R. H. C., & Fleming, P. J. (2010) Local search with Quadratic Approximation in Genetic Algorithms for expensive optimization problems. *IEEE Congress on Evolutionary Computation*, 677-683.

Weighted sum method. (n.d.). Retrieved May 20, 2105, from http://brasil.cel.agh.edu.pl/~13sustrojny/en/weighted-sum-method/

Wood, A. J., Woolenberg, B. F., & Sheble, G. B. (2013). *Power generation, operation and control* (3rd ed.). New York: John Wiley and Sons.

Wu, L. H., Wang, Y. N., Yuan, X. F., & Zhou, S. W. (2010). Environmental/economic power dispatch problem using multi-objective differential evolution algorithm. *Electric Power Systems Research*, *80*(9), 1171-1181.

Yang, X. S. (2010). *Engineering Optimization: An introduction with metaheuristic applications* (1st ed.). London: Wiley & Sons.

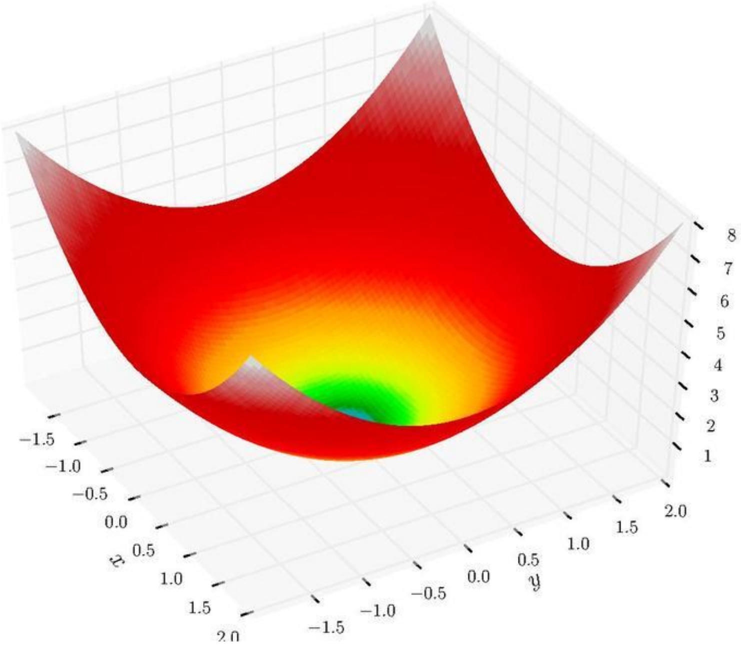
Yang, X. S. (2010). A new metaheuristic bat-inspired algorithm. *Nature Inspired Cooperative Strategies for Optimization (NISCO 2010)*, *284*, 65-74.

Zhang, Y., Gong, D-W., & Ding, Z. (2012). A bare-bones multi-objective particle swarm optimization algorithm for environmental/economic dispatch. *Information Sciences*, *192*, 213-227.

APPENDICES

APPENDIX A. Mathematical Representations and Plots of Benchmark Functions

- Sphere Function:

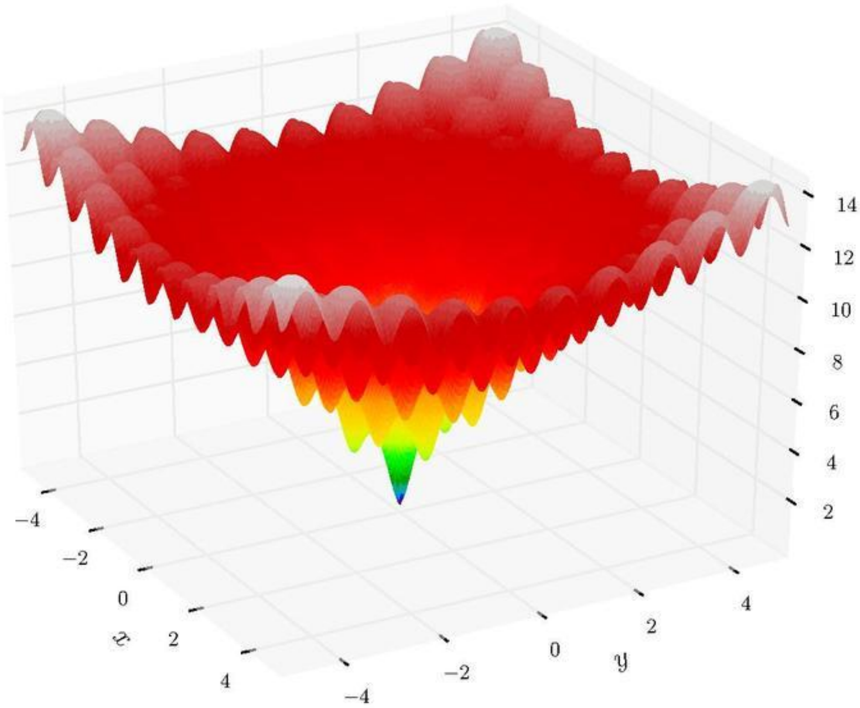


$$f(\mathbf{x}) = \sum_{i=1}^{d} x_i^2$$

$$f(\mathbf{x}^*) = 0, \text{ at } \mathbf{x}^* = (0, \dots, 0)$$

Search space: $x_i \in [-5.12, 5.12]$ for $i = 1,2,3,\dots,d$ in d-dimensional space

-Ackley Function:

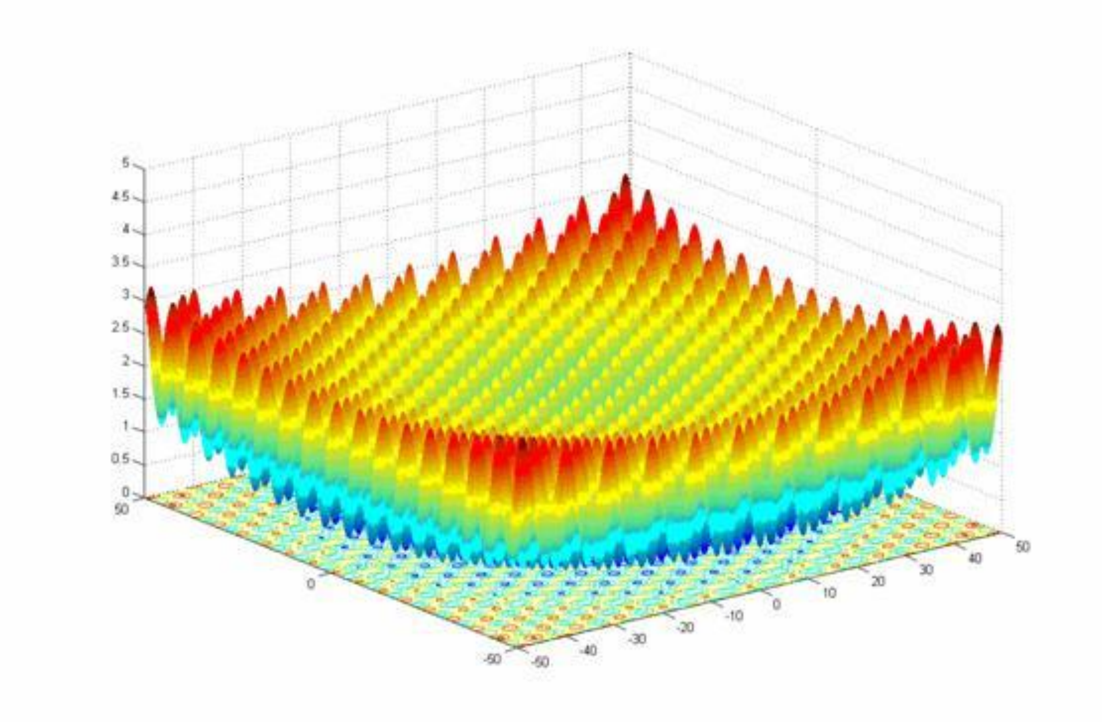


$$f(\mathbf{x}) = -a \exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(cx_i)\right) + a + \exp(1)$$

$$f(\mathbf{x}^*) = 0, \text{ at } \mathbf{x}^* = (0, \dots, 0)$$

Search space: $x_i \in [-32.768, 32.768]$ for $i = 1,2,3,\dots,d$ in d-dimensional space

-Griewank Function:

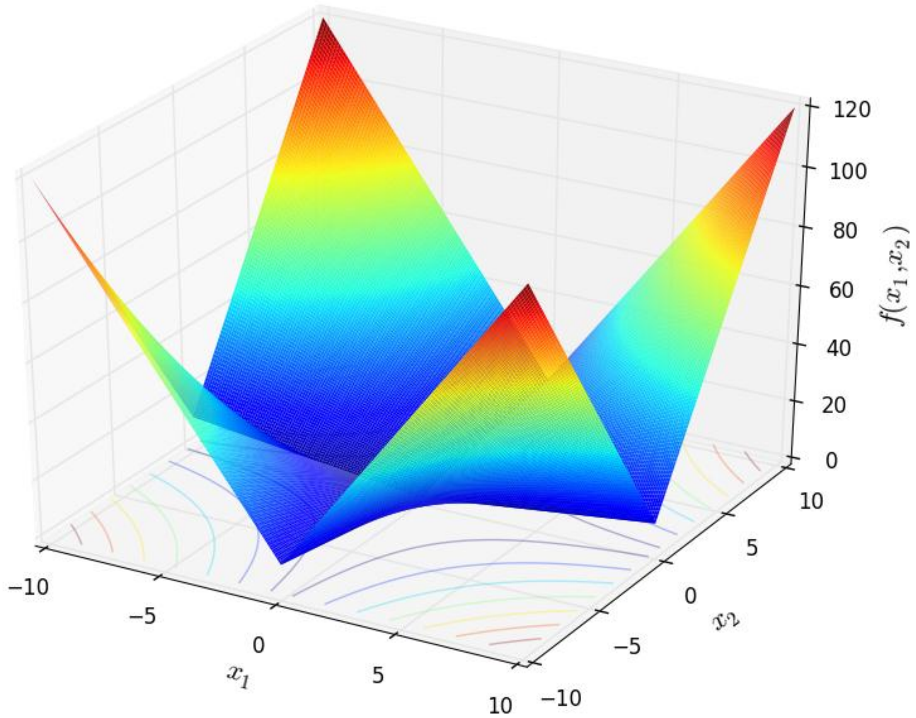


$$f(\mathbf{x}) = \sum_{i=1}^{d} \frac{x_i^2}{4000} - \prod_{i=1}^{d} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

$$f(\mathbf{x}^*) = 0, \text{ at } \mathbf{x}^* = (0, \dots, 0)$$

Search space: $x_i \in [-600, 600]$ for $i = 1,2,3,\dots,d$ in d-dimensional space

-Schwefel Function:

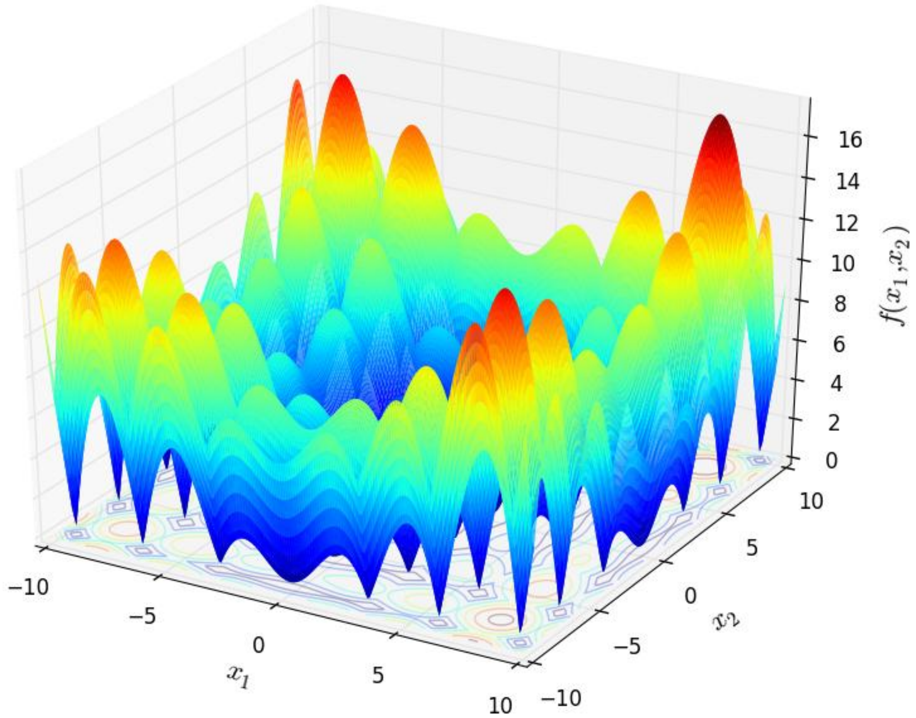


$$f_{\text{Schwefel22}}(\mathbf{x}) = \sum_{i=1}^{n} |x_i| + \prod_{i=1}^{n} |x_i|$$

Global optimum: $f\left(x_i\right) = 0$ for $x_i = 0$ for $i = 1,2,3,\ldots,d$

Search space: $x_i \in \left[-100,100\right]$ for $i = 1,2,3,\ldots,d$ in d-dimensional space

-Alpine Function:

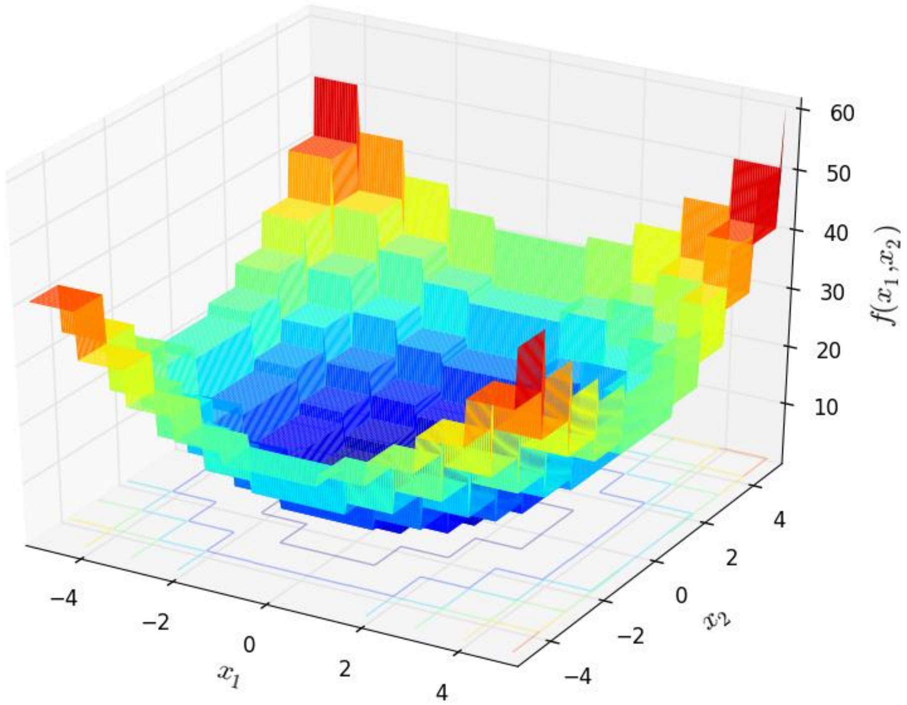


$$f_{\text{Alpine01}}(\mathbf{x}) = \sum_{i=1}^{n} \left| x_i \sin(x_i) + 0.1 x_i \right|$$

Global optimum: $f(x_i) = 0$ for $x_i = 0$ for $i = 1,2,3,\ldots,d$

Search space: $x_i \in [-10,10]$ for $i = 1,2,3,\ldots,d$ in d-dimensional space

-Step Function:



$$f_{\text{Step}}(\mathbf{x}) = \sum_{i=1}^{n} \left(\lfloor x_i \rfloor + 0.5\right)^2$$

Global optimum: $f\left(x_i\right) = 0$ for $x_i = 0.5$ for $i = 1,2,3,...,d$

Search space: $x_i \in \left[-100,100\right]$ for $i = 1,2,3,...,d$ in d-dimensional space

APPENDIX B. Cost and Emission Coefficients for the Generation Units

-6 Units System:

a=[10.0,10.0,20.0,10.0,20.0,10.0]

b=[200.0,150.0,180.0,100.0,180.0,150.0]

c=[100.0,120.0,40.0,60.0,40.0,100.0]

α=[4.091e-2,2.543e-2,4.258e-2,5.326e-2,4.258e-2,6.131e-2]

β=[-5.554e-2,-6.047e-2,-5.094e-2,-3.550e-2,-5.094e-2,-5.555e-2]

γ=[6.49e-2,5.638e-2,4.586e-2,3.380e-2,4.586e-2,5.151e-2]

η=[2.0e-4,5.0e-4,1.0e-6,2.0e-3,1.0e-6,1.0e-5]

δ=[2.857,3.333,8.000,2.000,8.000,6.667]

-10 Units System:

a=[1000.403,950.606,900.705,800.705,756.799,451.325,1243.531,1049.998,1658.569,1356.659]

b=[40.5407,39.5804,36.5104,39.5104,38.5390,46.1592,38.3055,40.3965,36.3278,38.2704]

c=[0.12951,0.10908,0.12511,0.12111,0.15247,0.10587,0.03546,0.02803,0.02111,0.01799]

d=[33.0,25.0,32.0,30.0,30.0,20.0,20.0,30.0,60.0,40.0] *e*=[0.0174,0.0178,0.0162,0.0168,0.0148,0.0163,0.0152,0.0128,0.0136,0.0141]

α=[360.0012,350.0056,330.0056,330.0056,13.8593,13.8593,40.2669,40.2669,42.8955,42.8955]

β=[-3.9864,-3.9524,-3.9023,-3.9023,0.3277,0.3277,-0.5455,-0.5455,-0.5112,-0.5112]

γ=[0.04702,0.04652,0.04652,0.04652,0.00420,0.00420,0.00680,0.00680,0.00460,0.00460]

η=[0.25475,0.25475,0.25163,0.25163,0.24970,0.24970,0.24800,0.24990,0.25470,0.25470]

δ=[0.01234,0.01234,0.01215,0.01215,0.01200,0.01200,0.01290,0.01203,0.01234,0.01234]

-11 Units System:

a=[387.85,441.62,422.57,552.50,557.75,562.18,568.39,682.93,741.22,617.83,674.61]

b=[1.92699,2.11969,2.19196,2.01983,2.22181,1.91528,2.10681,1.99138,1.99802,2.12352,2.10487]

c=[0.00762,0.00838,0.00523,0.00140,0.00154,0.00177,0.00195,0.00106,0.00117,0.00089,0.00098]

α=[33.93,24.62,33.93,27.14,24.15,27.14,24.15,30.45,25.59,30.45,25.59]
β=[-0.67767,-0.69044,-0.67767,-0.54551,-0.40060,-0.54551,-0.40006,-0.51116,-0.56228,-0.41116,-0.56228]

γ=[0.00419,0.00461,0.00419,0.00683,0.00751,0.00683,0.00751,0.00355,0.00417,0.00355,0.00417]

-40 Units System:

a=[94.705,94.705,309.54,369.03,148.89,222.33,287.71,391.98,455.76,722.82,635.20,654.69,913.40,1760.4,1760.4,1760.4,647.85,649.69,647.83,647.81,785.96,785.96,794.53,794.53,801.32,801.32,1055.1,1055.1,1055.1,148.89,222.92,222.92,222.92,107.87,116.58,116.58,307.45,307.45,307.45,647.83]

b=[6.73,6.73,7.07,8.18,5.35,8.05,8.03,6.99,6.60,12.9,12.9,12.8,12.5,8.84,8.84,8.84,7.97,7.95,7.97,7.97,6.63,6.63,6.66,6.66,7.10,7.10,3.33,3.33,3.33,5.35,6.43,6.43,6.43,8.95,8.62,8.62,5.88,5.88,5.88,7.97]

c=[0.00690,0.00690,0.02028,0.00942,0.0114,0.01142,0.00357,0.00492,0.00573,0.00605,0.00515,0.00569,0.00421,0.00752,0.00752,0.00752,0.00313,0.00313,0.00313,0.00313,0.00298,0.00298,0.00284,0.00284,0.00277,0.00277,0.52124,0.52124,0.52124,0.01140,0.0016,0.0016,0.0016,0.0001,0.0001,0.0001,0.0161,0.0161,0.0161,0.00313]

d=[100.0,100.0,100.0,150.0,120.0,100.0,200.0,200.0,200.0,200.0,200.0,200.0,300.0,300.0,300.0,300.0,300.0,300.0,300.0,300.0,300.0,300.0,300.0,300.0,300.0,300.0,120.0,120.0,120.0,120.0,150.0,150.0,150.0,200.0,200.0,200.0,80.0,80.0,80.0,300.0]

e=[0.084,0.084,0.084,0.063,0.077,0.084,0.042,0.042,0.042,0.042,0.042,0.042,0.035,0.035,0.035,0.035,0.035,0.035,0.035,0.035,0.035,0.035,0.035,0.035,0.035,0.035,0.077,0.077,0.077,0.077,0.063,0.063,0.063,0.042,0.042,0.042,0.098,0.098,0.098,0.035]
α=[60.0,60.0,100.0,120.0,50.0,80.0,100.0,130.0,150.0,280.0,220.0,225.0,300.0,520.0,510.0,510.0,220.0,222.0,220.0,220.0,290.0,285.0,295.0,295.0,310.0,310.0,360.0,360.0,360.0,50.0,80.0,80.0,80.0,65.0,70.0,70.0,100.0,100.0,100.0,220.0]

β=[-2.22,-2.22,-2.36,-3.14,-1.89,-3.08,-3.06,-2.32,-2.11,-4.34,-4.34,-4.28,-4.18, -3.34,-3.55,-3.55,-2.68,-2.66,-2.68,-2.68,-2.22,-2.22,-2.26,-2.26,-2.42,-2.42,-1.11, -1.11,-1.11,-1.89,-2.08,-2.08,-2.08,-3.48,-3.24,-3.24,-1.98,-1.98,-1.98,-2.68]

γ=[0.0480,0.0480,0.0762,0.0540,0.0850,0.0854,0.0242,0.0310,0.0335,0.4250,0.0322,0.0338,0.0296,0.0512,0.0496,0.0496,0.0151,0.0151,0.0151,0.0151,0.0145,0.0145,0.0138,0.0138,0.0132,0.0132,1.8420,1.8420,1.8420,0.0850,0.0121,0.0121,0.0121, 0.0012, 0.0012,0.0012 ,0.0950,0.0950,0.0950,0.0151]

η=[1.3100,1.3100,1.3100,0.9142,0.9936,1.3100,0.6550,0.6550,0.6550,0.6550,0.6550,0.6550,0.5035,0.5035,0.5035,0.5035,0.5035,0.5035,0.5035,0.5035,0.5035,0.5035,0.5035,0.5035,0.5035,0.5035,0.9936,0.9936,0.9936,0.9936,0.9142,0.9142,0.9142,0.6550,0.6550,0.6550,1.4200,1.4200,1.4200,0.5035]

δ=[0.05690,0.05690,0.05690,0.04540,0.04060,0.05690,0.02846,0.02846,0.02846,0.02846,0.028460,0.02846,0.02075,0.02075,0.02075,0.02075,0.02075,0.02075,0.02075,0.02075,0.02075,0.02075,0.02075,0.02075,0.02075,0.02075,0.04060,0.04060,0.04060,0.04060,0.04540,0.04540,0.04540,0.02846,0.02846,0.02846,0.06770,0.06770,0.06770,0.02075]