

Abstract

BİRİL, Ş. İLKER. Stochastic Global Optimization Techniques. (Under the direction of Shu-Cherng Fang)

In this research, a novel population-based global optimization method has been studied. The method is called Electromagnetism-like Mechanism or in short EM. The proposed method mimicks the behavior of electrically charged particles. In other words, a set of points is sampled from the feasible region and these points imitate the role of the charged particles in basic electromagnetism. The underlying idea of the method is directing sample points toward local optimizers, which point out attractive regions of the feasible space.

The proposed method has been applied to different test problems from the literature. Moreover, the viability of the method has been tested by comparing its results with other reported results from the literature. Without using the higher order information, EM has converged rapidly (in terms of the number of function evaluations) to the global optimum and produced highly efficient results for problems of varying degree of difficulty.

After a systematic study of the underlying stochastic process, the proof of convergence to the global optimum has been given for the proposed method. The thrust of the proof has been to show that in the limit, at least one of the points in the population moves to the

neighborhood of the global optimum with probability one.

The structure of the proposed method is very flexible permitting the easy development of variations. Capitalizing on this, several variants of the proposed method has been developed and compared with the other methods from the literature. These variants of EM have been able to provide accurate answers to selected problems and in many cases have been able to outperform other well-known methods.

STOCHASTIC GLOBAL OPTIMIZATION TECHNIQUES

BY

Ş. İLKER BİRBİL

A DISSERTATION SUBMITTED TO THE GRADUATE FACULTY OF

NORTH CAROLINA STATE UNIVERSITY

IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

DEPARTMENT OF INDUSTRIAL ENGINEERING

RALEIGH

MARCH, 2002

APPROVED BY:

SHU-CHERNG FANG

CHAIR OF ADVISORY COMMITTEE

HENRY L. W. NUTTLE

YAHYA FATHI

ELMOR L. PETERSON

XIULI CHAO

*To Pınar,
my absolute point of attraction...*

Biography

Ş. İlker Birbil was born on December 29th, 1973 in Istanbul, Turkey. In 1995, he received his B.Sc. degree in Industrial Engineering from Yıldız Technical University in Istanbul. He started his M.Sc. degree in Marmara University, Department of Industrial Engineering. Then he transferred to Yeditepe University, Department of Systems Engineering and he completed his M.Sc. degree in 1997. The same year he started his Ph.D. degree in Boğaziçi University in Istanbul. In 1999, after a radical change in his life, he decided to follow a complicated path and started his doctorate study in North Carolina State University, Industrial Engineering Department. He received his Ph.D. degree in 2002 with a major in Industrial Engineering and minors in Operations Research and Mathematics. Currently, he is seeking for another complicated path.

Acknowledgment

I thank my advisor Professor Shu-Cherng Fang for his invaluable help. I learned how to build my own tree from him. I am also indebted to Professor Henry L. W. Nuttle, who has always been very kind and supportive - even when I knock on his door with a pile of paperwork in my hand. I gratefully acknowledge the comments and the encouragement of Professor Yahya Fathi. I felt very lucky to have Professor Elmor L. Peterson in my committee. I consider him as my grand advisor and I thank him for his challenging questions. I also thank Professor Xiuli Chao for his careful suggestions.

Many thanks go to Professor Salah Elmaghraby for his constant support and encouragement. Professor Ruey-Lin Sheu helped me greatly for the rigorous mathematics, I am grateful to him. I also thank Professor Jim Wilson for being an excellent department head and I acknowledge the help of the staff of Industrial Engineering Department and Operations Research program.

I thank my officemates. I have always considered myself very lucky to be in the same office with them. I thank Syhy-Huei Chen for being my coffee partner. Whenever I needed a break to walk around the campus, Hao Cheng was always there to accompany me. I have always enjoyed our discussions with Razek Karnoub about politics and science. Saowanee Lertworasirikul (Ma'am), Yi Liao and I have defended our theses in the same week, and

hence we shared the same anxiety and excitement. I thank both of them for their invaluable friendship. In numerous occasions, Yue Dai cheered me up with her joy. I am indebted to her. I thank Shunmin Wang for his generous help and for his admirable honesty. I also thank other members of the Fangroup, especially I thank Carin Lightner, Negar Arefi, and Burcu Özçam for their encouragement.

During my doctorate study, I have had happy as well as hard times. In any case I have felt comfortable when I picked up a play from the shelf. I am grateful to many playwrights for their wonderful plays. In particular, I take this opportunity to thank Samuel Beckett, Edward Albee, Bertolt Brecht, and Harold Pinter.

I have shared many things with my close friends and I have always sought for their support. I thank Mine Haşhaş for being a wonderful friend and roommate. I also thank Şeref Girgin for his constant support. He has always been a brother to me. Finally, my deepest thanks go to Pınar Yolum, my mother, Emel Kalkavan and my sister, İlkay Birbil (soon to be Raoul). Their love has been the most precious thing in my life.

Table of Contents

List of Tables	ix
List of Figures	x
1 Introduction	1
1.1 The Problem Statement	1
1.1.1 Importance of Global Optimization	3
1.1.2 Difficulties In Solving Global Optimization Problems	5
1.2 Contributions of This Research	7
1.3 Outline of The Dissertation	8
2 Literature Review	10
2.1 Taxonomy of Global Optimization Methods	10
2.2 Deterministic Methods	12
2.3 Stochastic Methods and Heuristics	15
2.3.1 Random Search Methods	16
2.3.2 Heuristic Strategies	18
2.4 Other Work	20
3 Electromagnetism-like Mechanism (EM)	22
3.1 General Scheme for EM	23
3.1.1 Initialization	24
3.1.2 Local Search	25
3.1.3 Calculation of Total Force Vector	27
3.1.4 Movement According to Total Force Vector	29
3.1.5 Termination Criteria	30
3.2 Computational Results	31
3.2.1 General Test Functions	32
3.2.2 Dixon and Szegö Test Functions	36

3.3	Effects of Different Parameters	39
3.4	Summary	41
4	Theoretical Structure	43
4.1	Modifications on the Original Method	44
4.1.1	Premature Convergence	44
4.1.2	Refined General Scheme	44
4.1.3	The Perturbed Point and The Modified <i>CalcF</i> Procedure	46
4.2	Convergence Results for the Modified EM	47
4.2.1	Notation and Assumptions	48
4.2.2	Convergence With Probability One	50
4.2.3	Computational Considerations	59
4.3	Computational Results	59
4.3.1	Dixon and Szegö Test Functions	60
4.3.2	Hard Test Functions	61
4.4	Summary	62
5	Variants of EM and Extensions	63
5.1	Using Higher Order Information	64
5.2	EM for Nonsmooth Convex Minimization: EMNCM	67
5.2.1	Some Simplifications	67
5.2.2	Comparison with Downhill Simplex Method	69
5.3	EM with Adaptive Step Length: EMASL	71
5.3.1	Change In Move Procedure	71
5.3.2	Comparison with a Simulated Annealing Variant	71
5.4	Constrained Optimization Problems	73
5.4.1	Penalty and Barrier Methods	76
5.4.2	Computational Study	77
5.5	Summary	80
6	Conclusion and Further Research	82
6.1	Overview of Accomplishments	82
6.2	Future Research Directions	83
	Bibliography	86
A	An Example Run of EM	99

B	Test Functions	103
B.1	General Test Functions	103
B.2	Dixon and Szegö Test Functions	108
B.3	Hard Test Functions	111
B.4	Nonsmooth Convex Test Functions	112
B.5	Constrained Test Problems	114

List of Tables

3.1	Parameters for General Test Functions	32
3.2	Results of EM without <i>Local</i> procedure	33
3.3	Results of EM with <i>Local</i> procedure applied to all points	34
3.4	Results of EM with <i>Local</i> procedure applied to current best point	35
3.5	Results of EM for Dixon and Szegö [1975] test functions	36
3.6	Comparison of EM with different methods in terms of number of function evaluations	38
3.7	Levels of the factors used in experimental design	40
3.8	Analysis of variance table for the function <i>Sine Envelope</i>	40
3.9	Analysis of variance table for the function <i>Shekel</i> [S5]	41
4.1	Results for Dixon and Szegö [1975] test functions with the refined algorithm	61
4.2	Results for hard test functions	62
5.1	Results of EM using higher order information for Dixon and Szegö [1975] test functions	65
5.2	Comparison of EM using higher order information with other methods for Dixon and Szegö [1975] test functions	66
5.3	Comparison of EMNCM and downhill simplex method	70
5.4	Comparison of EMASL with SA for general test functions	74
5.5	Comparison of EMASL with SA for Dixon and Szegö [1975] test functions	75
5.6	Parameters for constrained test problems	79
5.7	Comparison of penalty and barrier methods on constrained test problems . .	79

List of Figures

1.1	Illustration of difficulties in optimizing a nonlinear function.	6
2.1	The taxonomy of global optimization methods	11
3.1	Superposition Principle ($\vec{F}_{i3} = \frac{q_3q_i}{4\pi\epsilon_0\epsilon_r r^2} \vec{e}_r$ $i = 1, 2$).	27
4.1	An example of premature convergence on one dimensional space	45
4.2	Truncated cone in \mathfrak{R}^2	50
4.3	Calculation of the probability for Step 2 in \mathfrak{R}^2	52
4.4	Calculation of the lower bound for $\mu(T(\mathbf{x}))$ in \mathfrak{R}^2	54
4.5	The calculation of r^* in \mathfrak{R}^2	56
A.1	<i>Goldstein Price</i> function	100
A.2	The starting position of the particles just after the procedure <i>Initialize</i>	100
A.3	A new point with better objective value.	101
A.4	Points start to attract and repel each other.	101
A.5	Optimum is found. Points move toward the current best point	102

Chapter 1

Introduction

In recent years global optimization has become a rapidly developing field. This is because global optimization problems arise in diverse application areas. Despite its importance and the efforts invested so far, the methods for solving global optimization problems are not satisfactory enough. Therefore, the challenge of determining an absolutely best solution for a general nonlinear function with many variables and complex attributes attracts the attention of researchers.

1.1 The Problem Statement

We consider the following global optimization problem:

Given a nonempty set $S \subset \mathfrak{R}^n$ and a real-valued function $f : S \rightarrow \mathfrak{R}$, find at least one point $x^* \in S$ such that $f(x^*) \leq f(x)$ for all $x \in S$.

Following this statement, we denote a global optimization problem¹ by

$$\begin{aligned} & \text{minimize} && f(x) \\ & \text{subject to} && x \in S, \end{aligned} \tag{1.1}$$

where f is the *objective function* and S refers to the *feasible region*. A vector $x^* \in S$ satisfying $f(x^*) \leq f(x)$ for all $x \in S$ is called a *global minimizer* of f over S and the corresponding value of f is called a *global minimum*.

Let $\|\cdot\|$ denote the Euclidean norm in \mathfrak{R}^n and let $\varepsilon > 0$ be a real number. Then an ε -*neighborhood* of a point $y \in \mathfrak{R}^n$ is defined as the open ball

$$B(y, \varepsilon) \triangleq \{x \in \mathfrak{R}^n : \|x - y\| < \varepsilon\}. \tag{1.2}$$

A point $x^* \in S$ is called a *local minimizer* of f over S , if there is an $\varepsilon > 0$ such that

$$f(x^*) \leq f(x), \forall x \in B(x^*, \varepsilon) \cap S. \tag{1.3}$$

Notice that by definition every global minimizer is a local minimizer, but the converse is not true in general.

Throughout this research, we study a special class of optimization problems with bounded variables in the form of (1.1) where

$$S \triangleq \{x \in \mathfrak{R}^n \mid l_k \leq x_k \leq u_k, \ l_k, u_k \in \mathfrak{R} \text{ for } k = 1, \dots, n\}. \tag{1.4}$$

We recall that, when f is continuous and S is a compact subset of \mathfrak{R}^n , the existence of x^* is assured by the Weierstrass theorem of the classical analysis [Taylor and Mann, 1972, Horst and Tuy, 1993]. Because of the ease of maintaining feasibility, the feasible region defined in (1.4) has been extensively studied by different classes of global optimization methods. A rigorous discussion on different methods is given in Chapter 2.

¹In the rest of the dissertation we deal with minimization problems, since maximization problems can be easily converted into minimization problems.

1.1.1 Importance of Global Optimization

The idea of finding the optimum solution of a problem goes back to the development of calculus and is associated with the names of Newton, Lagrange, and Cauchy. Particularly after World War II, numerous theoretical and computational methods were proposed for solving optimization problems. Most of these methods are capable of solving problems that have relatively simple structure, i.e., problems that include functions of few variables or that include linear or usually unimodal functions. However, a large variety of problems that include multimodal nonlinear functions with many variables, arise in different application areas. Some of these application areas are engineering design, production management, computational chemistry, product mixture design, environmental pollution management, parameter estimation, VLSI design, and neural network learning [Floudas, 2000]. Therefore finding the optimum solution of an arbitrary function becomes an important challenge.

In last three decades many researchers from different fields such as Applied Mathematics, Operations Research, Industrial Engineering, Management Science, Computer Science, and so on, study global optimization to meet this challenge. This diversity stems from the fact that global optimization covers a wide range of mathematical optimization methods. Specifying special properties of the objective function f and/or the feasible region S in (1.1), leads to a different problem that requires a different approach [Pintér, 1996a, Floudas, 2000]:

- Concave minimization: f is a concave function and S is a convex set.
- D.C. programming: f can be represented as difference of two convex functions and S consists of equalities and inequalities that can be represented as difference of two convex functions.
- Bilinear and biconvex programming: f is a bilinear or a biconvex function and

S is a convex set.

- Combinatorial optimization: f is defined on discrete points, and S consists of discrete points.
- Network problems: f is an arbitrary function, and S usually consists of linear or convex equilibrium constraints.
- Quadratic global optimization: f is an arbitrary quadratic function, and S consists of linear (or quadratic) equalities and inequalities.
- Lipschitz optimization: f is an arbitrary Lipschitz continuous function, and S consists of inequalities that are represented by Lipschitz continuous functions.
- Linear and nonlinear complementarity problems: f is product of two vector functions, and S is usually a convex set.
- Generalized geometric programming problems: f belongs to a special class of functions called *signomials*, S consists of equalities and inequalities that are represented by signomials.
- Fractional programming: f is a ratio of two real-valued functions, S is convex.
- Nonlinear programming problems: Usually f belongs to the class of twice continuously differentiable functions, and S is a compact set.
- Multiplicative programming: f is represented as the product of several convex functions, and S consists of equalities or inequalities that are represented by convex functions or functions that are product of convex functions.
- Seperable programming: f is an arbitrary seperable function, and S is usually a convex set.
- Linear programming: f is a linear function, and S is a convex set defined by system of linear equalities and inequalities.

- Other mathematical programming problems including convex programming, parametric optimization, and stochastic optimization.

A quick glance at the rich literature on global optimization shows its significant growth. We refer to the book by Floudas [2000] for a representative list of publications and conference proceedings.

1.1.2 Difficulties In Solving Global Optimization Problems

It is well-known that classical gradient-based methods such as the Quasi-Newton method [Gill and Murray, 1972, Fletcher and Reeves, 1964], modified steepest descent method [Armijo, 1996], or conjugate gradient method [Fletcher and Reeves, 1964] may be trapped by local optima when the feasible region around the global optimum is not well-conditioned [Beveridge, 1970]. This is because all standard tools of nonlinear optimization such as derivatives, gradients, subgradients and the like, can at most determine local minima. In other words, unless certain restrictive assumptions like convexity, Lipschitz condition, etc. are satisfied, global minimizers are not only hard to find, but also hard to verify.

In order to develop efficient methods, it is important to study the difficulties faced by the classical methods [Shang, 1997]. Let us illustrate some of these difficulties on a multimodal nonlinear function in Figure 1.1.

- The global minimizer A is in a deep valley and it is surrounded by tall hills that are difficult to overcome by gradient-based methods.
- B is a promising local minimizer that is in the neighborhood of the global minimizer. A search method may be trapped in B before reaching A.
- A search method may spend excessive iterations before passing the shallow basin C.

- The promising local minimizer D may mislead the search method to the regions that are far from the global minimizer.
- The function is nondifferentiable at E, hence the classical gradient-based methods can not be applied.

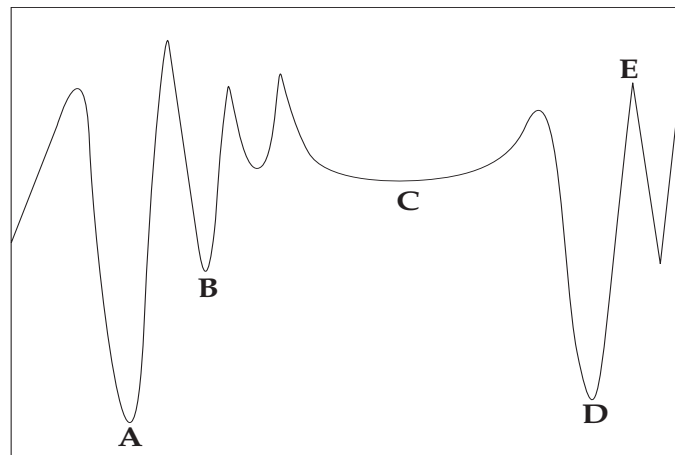


Figure 1.1: Illustration of difficulties in optimizing a nonlinear function.

The complexity of solving global optimization problems has been already shown in a number of theoretical studies. In particular, Murty and Kabadi showed that there exist quadratic and nonlinear programming problems that belong to the class of NP-complete problems [1987]. Rinooy Kan and Timmer proved that global optimization problems are not solvable in finite number of steps [1989]. A similar result has been shown by Pardalos and Vavasis for quadratic programming problems [1991].

1.2 Contributions of This Research

In this dissertation, we introduce a new stochastic global optimization method that is able to overcome some of the shortcomings of other approaches. We also provide a proof of convergence to the global optimum for the proposed method.

To be more specific:

- The proposed method does not require restrictive assumptions like differentiability, convexity, etc. It works without the higher order information. This permits us to focus on solving multi-modal, irregular, high dimensional functions, which are difficult to solve by conventional numerical methods.
- Like many multi-point methods, the proposed method also works on a set of sample points (*population*) of the feasible region and proceeds according to relative efficiencies of the observed functional values. However, the structure of the encoding is specifically developed for continuous optimization problems. Thus, the method can be much faster and the number of function evaluations can be far less than other methods.
- To intensify the search, we use an *attraction* mechanism between the sample points rather than partitioning the feasible domain. We add a diversification effect by using a *repulsion* mechanism among the points.
- Unlike the typical multi-start techniques, the sample points are related to each other. In other words, the choice of a sample point in successive iterations depends on the computational experience of the other points in the population.
- We compare our results with other global optimization methods. In order to come up with a competitive method, we pay particular attention to creating a

flexible, and at the same time, efficient structure. This enables us to elevate the performance of our approach with additional tools provided by the other methods.

- We prove that the proposed method converges with probability one. The structure of the proof is flexible and can be generalized for the study of convergence of other population-based methods.
- In constrained optimization problems, the proposed method may be used as a preprocessing tool for generating feasible points from complicated regions.
- The general structure of the proposed method makes it straightforward to develop parallel algorithms.

1.3 Outline of The Dissertation

Current chapter is followed by Chapter 2, which includes a literature survey on different global optimization methods. The pros and cons of these methods are also discussed in Chapter 2.

Chapter 3 begins with the motivation behind the proposed method and continues with the description of the essential procedures. After the presentation of the proposed method, computational results on different problems are given. Chapter 3 ends with an experiments section that explains the effects of different parameters.

In Chapter 4, we show the convergence properties of the proposed method. In order to achieve the desired result, several procedures are modified. Then it is shown that the proposed method converges with probability one. The chapter ends with a discussion on the computational experience with the modified algorithm.

Several variants of the proposed method along with some extensions are given in Chapter 5. Each variant is tested on a set of problems and then compared with some other methods. Chapter 5 also covers a preliminary study regarding constrained optimization problems.

We conclude the dissertation with Chapter 6. In addition to giving an overview of the dissertation, we discuss further research areas in a separate section. The appendices contain an example run of the proposed method and the test functions used throughout the dissertation.

Chapter 2

Literature Review

Dixon and Szegö published one of the earliest collection of papers devoted to global optimization [1975, 1978]. In light of this pioneering work, considerable variety of global optimization models and solution approaches have been studied. Here, we give a brief review of the global optimization literature.

2.1 Taxonomy of Global Optimization Methods

In the field of global optimization, it is quite common to classify the methods into two main categories: deterministic methods and stochastic (or probabilistic) methods [Törn and Zilinskas, 1989, Horst and Pardalos, 1995]. As the names imply, the methods that do not involve random elements go into the former category, whereas the methods that utilize probabilistic schemes go into the latter category. We remark that for many global optimization methods, the distinction between deterministic and stochastic is blurred. There are very effective methods, which involve deterministic as well as stochastic elements.

In recent years, with the increase in computer speed, a third category - heuristics - has

emerged. Many of these heuristic strategies involve stochastic elements. Hence we extend the stochastic methods category to include the heuristic methods.

This review consists of two parts: The first part presents the deterministic global optimization methods. The second part gives a comprehensive treatment of stochastic methods and heuristics - in which our proposed method also appears. Beyond our discussion, many additional references for specific approaches can be found within the cited literature.

Figure 2.1 gives the taxonomy for global optimization methods that we shall follow in our review.

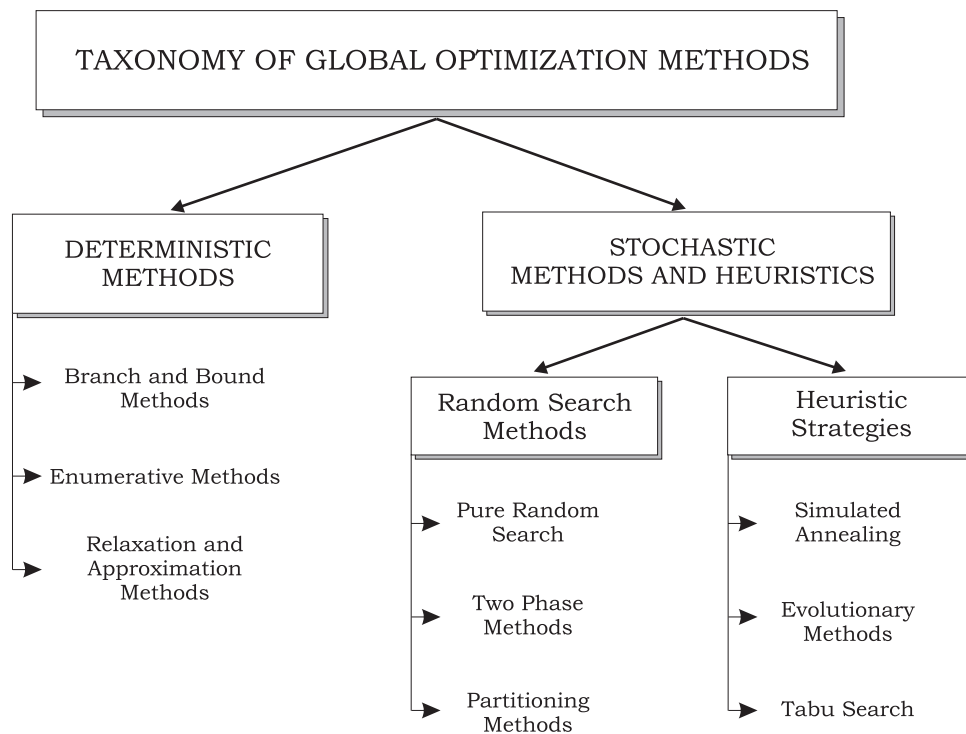


Figure 2.1: The taxonomy of global optimization methods

2.2 Deterministic Methods

In this section, we introduce some of the deterministic global optimization methods. The methods discussed in this section are direct methods, for which strong theoretical convergence results can be proven under certain assumptions. Most of these methods are developed for special classes of problems. Hence their performances may deteriorate with different problem classes. In addition, a method with a global convergence property may be so complicated that it may be intractable from a practical point of view.

Surveys and monographs which provide general coverage of deterministic methods include Dixon and Szegö [1975], Neumaier [1990], Floudas and Pardalos [1992], Horst and Tuy [1993], Kearfott [1996], Pintér [1996b], Floudas [2000], and Horst et al. [2000].

Branch and Bound Methods

Branch and bound methods are based on the idea of decomposing the original problem into smaller subproblems (*branches*) that are easier to handle. The same process is applied to the subproblems, and this process goes on until the optimal solution of any subproblem provides an optimal solution to the original problem. Nevertheless, as the number of branches increases, the number of subproblems grows exponentially. Hence, it becomes crucial to eliminate some of the subproblems. This requires a *bounding* scheme, which is based on setting lower bounds on the optimal objective function values of the subproblems.

Initially, branch and bound methods were developed for integer programming problems, where the exhaustive search feature of the methods guarantees the global convergence [Nemhauser and Wolsey, 1988]. Recently, branch and bound methodology has been successfully applied to other global optimization problems [Horst, 1986]. For example,

Pardalos and Rosen [1986] give a survey on the applications of branch and bound methods to solve general concave minimization problems.

Branch and bound methods are also applied to continuous global optimization problems, where the exhaustive enumeration concept is replaced by a global convergence argument [Horst and Tuy, 1987, Pintér, 1992, Csendes and Pintér, 1993, Kearfott, 1996]. Lipschitz optimization and interval methods are two effective approaches along this line [Ratschek and Rokne, 1995, Hansen and Jaumard, 1995]. These methods are extensively used for solving real world applications [Pintér, 1996b]. Nevertheless, both methods require a priori knowledge relative to the structure of the problem. In Lipschitz optimization, it is assumed that the modeler knows suitable constants (namely Lipschitz constants), which show how rapidly each function vary [Armijo, 1996, Pintér, 1996b]. In interval methods the functions are required to satisfy certain smoothness properties [Neumaier, 1990, Ratz and Csendes, 1995, Kearfott, 1996].

Enumerative Methods

In particular cases, a global optimization problem may have finite number of local minimizers. Enumerative methods try to find all these minimizers in order to locate the global minimizer. This seemingly “elementary” idea is supported by rigorous theoretical results and many successful implementations [Watson and Yang, 1980, Diener, 1986].

Path-following (or trajectory) methods rely on constructing a set of paths (or trajectories) such that all solutions of the problem are known *a priori* to lie on one of these paths. In many cases, these paths are derived from solving a system of differential equations. Starting from an initial point, the solutions along these paths are usually found by a numerical approximation method. This step is called *tracing* [Diener, 1995].

Homotopy (or parameter continuation) methods are similar to path-following methods. Homotopy methods work by first solving a simple problem and then parametrically transforming this problem into the original problem. This transformation leads to the concept of homotopies, which are used for generating the set of paths. After finding the paths to be traced, this approach can be extended to search for all solutions of the problem [Garcia and Zangwill, 1981, Forster, 1995].

The main drawback of the path-following and homotopy methods is the lack of a guarantee of generating all the paths leading to the solutions. Another restriction common to these methods is the requirement that the functions to be twice continuously differentiable.

Relaxation and Approximation Methods

The history of relaxation and approximation methods precedes the introduction of global optimization as a separate field. In particular, these methods were used for solving difficult combinatorial optimization problems [Nemhauser and Wolsey, 1988].

The essential idea in relaxation and approximation methods is to solve a sequence of simple subproblems, which are derived from the original problem by successive relaxations. Then, these subproblems are extended by adding constraints such that the original problem is more closely approximated [Horst and Tuy, 1993, Horst and Pardalos, 1995].

A very important approach among these methods is the cutting plane method [Kelley, 1960, Westerlund and Pettersson, 1995]. The basic idea of the cutting plane method is to iteratively cut off parts of the feasible region that does not include the global minimizer. Then the search for the solution of the problem continues in the remaining part of the feasible region. It has been shown that cutting plane methods are useful for diverse global optimization problems such as, concave minimization [Benson, 1995], differential convex

(D.C.) and reverse convex problems [Tuy, 1987b,a, Thoai, 1988].

There are different difficulties that may arise with relaxation and approximation methods. Mainly, the subproblems derived from the original problem can still be complex. In this case, a further relaxation may cause too much oversimplification of the original problem. Furthermore, adding a constraint at each iteration may complicate the subproblems so that an efficient constraint dropping strategy may become necessary.

2.3 Stochastic Methods and Heuristics

This section covers global optimization methods that are mostly based on random sampling from the feasible region. Some of these stochastic methods converge (with probability one) to the global minimizer. We also review some of the heuristic strategies, which are motivated by certain analogies with natural phenomena or basic sciences. In many cases, these heuristics provide usable solutions even when deterministic methods fail because of the irregularities or high dimensionality. However, for most of the heuristics a rigorous treatment of global convergence properties does not exist.

For a general coverage of different stochastic methods and heuristics, see Dixon and Szegö [1978], Pintér [1984], Boender et al. [1985], Rinooy Kan and Timmer [1987a, 1987b], Laarhoven and Aarts [1987], Törn and Zilinskas [1989], Goldberg [1989], Zimmermann [1990], Zhigljavsky [1991], Kosko [1993], Rinooy Kan and Timmer [1994], Boender and Romeijn [1995], Osman and Kelly [1996], Aarts and Lenstra [1997], and Glover and Laguna [1997].

2.3.1 Random Search Methods

Random search methods constitute the core element of many complex stochastic global optimization methods. In this section, we try to include the methods that have played a pioneering role in the development of many other stochastic methods.

Pure Random Search

Pure random search is the simplest stochastic method for solving global optimization problems. Main idea is to sample a sequence of independent identically distributed random points from the feasible region, while keeping the track of the best objective function value achieved [Brooks, 1958, Anderssen and Bloomfield, 1975, Zhigljavsky, 1991, Zabinsky and Smith, 1992]. In this case, the proof of convergence to the global optimum (with probability one) can be easily shown [Baba et al., 1977, Devroye, 1978, Baba, 1981, Solis and Wets, 1981, Pintér, 1984].

In general, pure random methods are easy to implement. Therefore, they are often used as a first approach to solve complex problems. Moreover, in many cases these methods are modified leading to more complicated algorithms [Smith, 1984, Berbee et al., 1987]. However, pure random search methods are not efficient, especially for problems with higher dimensions, since in this case the number of iterations required to solve a problem increases fast.

Two-Phase Methods

In two-phase methods, the search is divided into two steps. In the *global phase*, the function is evaluated at a number of randomly sampled points, while in the *local phase*, the sample points are manipulated by means of local search to yield a candidate global minimum

[Rinooy Kan and Timmer, 1987a, Boender and Romeijn, 1995].

Density Clustering, Controlled Random Search, Multistart, and the Multi Level Single Linkage (MLSL) [Rinooy Kan and Timmer, 1994] are well-known two-phase methods that enable the exploration of the whole feasible region through random sampling followed by hill-climbing or gradient-based methods. In spite of a few successful results [Boender et al., 1985], there are numerous cases when these methods are inefficient. The main reason for this is that they inevitably cause each local minimum to be visited several times [Dekkers and Aarts, 1991].

Topographical MLSL (TMLSL) [Ali and Storey, 1994] and Topographical Optimization with Local Search (TOLS) [Törn and Viitanen, 1994] are two more recently developed methods of this type. TMLSL and TOLS are both reported to be considerably superior to MLSL and other previously introduced clustering methods. Nevertheless, these methods may fail in two different ways. First, the resulting groups of points, or clusters, may contain several regions of attractions, so that the global minimum can be missed. Second, one region of attraction may be divided over several clusters, in which case the corresponding optimum will be located more than once [Rinooy Kan and Timmer, 1987a].

Partitioning Methods

Partitioning schemes, which divide the feasible region into smaller subspaces, discard some unfavorable regions, and re-partition promising areas to reduce the final search space, have also been proposed for global optimization [Jones et al., 1993, Wood, 1991, Caprani and Madsen, 1993].

Recently, Demirhan et al. proposed a partitioning algorithm, with a fuzzy region selection criterion [1999]. Promising results are obtained on 13 test functions obtained from the

literature [Anroulakis and Vrahatis, 1996, Michalewicz, 1994, Srinivas and Patnaik, 1994].

Difficulties arise when the function has attractive regions spread around the feasible space. In this case, partitioning methods may discard an area which actually includes the global optimum. In addition, in most cases an extensive number of function evaluations is inevitable.

2.3.2 Heuristic Strategies

Starting in the 1980's, a wide range of novel approaches, which rely heavily on computer speed, have been developed. Some of these methods are inspired by careful observations of natural phenomena and some of them are developed merely by implementing practical ideas.

Simulated Annealing

Simulated Annealing (SA) is a stochastic single-point search technique, which has been applied successfully in various optimization fields. SA guides the search by specifying a *cooling scheme* that allows the acceptance of randomly generated neighbor solutions which are relatively unfavorable as compared with the current solution [Kirkpatrick et al., 1983, Laarhoven and Aarts, 1987]. Using the latter feature, SA aims to escape entrapment at local optima. The *temperature* described by the cooling scheme is reduced as the search makes progress so that the search is intensified around the global optimum. There exist different convergence proofs for the variants of SA [Dekkers and Aarts, 1991, Boender and Romeijn, 1995]. The strong points of SA and some pitfalls for potential SA users are indicated in an extensive review given by Ingber [1994], where a wide range of application areas from finance to combat analysis are described. Parallelization of the SA procedure as

a multi-start search technique is also discussed in the same paper.

In spite of the successful results reported for SA [Kirkpatrick et al., 1983, Aarts and Korst, 1997], a serious deficiency in the method is that the results depend on the starting point of each run. Furthermore, any combination with a multi-start technique and/or application of parallelization techniques increases the computational effort drastically [Birbil et al., 1999, Özdamar and Demirhan, 2000].

Evolutionary Methods

Although initially described as classifier systems and reflections of evolutionary systems, Genetic Algorithms (GA) have recently become popular global optimization techniques, specifically in combinatorial optimization [Goldberg, 1989]. GAs investigate the feasible space by utilizing populations of solutions, which evolve by reproduction, crossover, and mutation operators [Michalewicz, 1994].

An interesting method using GA operators is provided by Wodrich and Bilchev [Wodrich and Bilchev, 1997]. The authors use crossover and mutation operators to create new regions of search for the Ant Colonization technique originally proposed for the Traveling Salesman Problem by Dorigo and Coloni [Dorigo and Coloni, 1996]. In their case, all of numerical results are for combinatorial optimization problems. To our knowledge, all the implementations of Ant Colonization technique have been for discrete optimization problems. Nevertheless, it is reasonable to expect new implementations for continuous optimization inspired by the discrete counterparts.

Although they provide powerful optimization tools for difficult optimization problems [Schaffer, 1989, Birbil, 1999], a serious handicap of evolutionary methods is the heavy computational effort required to solve large scale problems [Demirhan et al., 1999]. In

addition, the structure of the encoding used in these methods is not usually adequate for the continuous global optimization problems.

Tabu Search

The first influential work on tabu search was published in late 1980's by Glover [1986]. After this pioneering result, additional efforts on formalization [Glover, 1989, 1990] and a monograph for a complete discussion were presented [Glover and Laguna, 1997]. Furthermore, a recent work by Glover et al. showed that tabu search is a flexible approach that can be merged with other methods [1995].

Tabu search is an iterative scheme, which is based on moving from one point to a neighborhood point in single iteration. Throughout this search, main idea is to keep track of not only the local information but also some information related to the exploration process. This scheme requires a systematic handling of (tabu) lists, which hold the history of moves and prevents them to be revisited.

Though no rigorous convergence proof has been shown, many successful applications of tabu search are reported [Glover and Laguna, 1997]. However, the successful applications are given only for combinatorial optimization problems and the applications for continuous global optimization problems are in their early stages of development [Battiti and Tecchiolli, 1996].

2.4 Other Work

Before describing our proposed approach, we cite several references we used in our computational work. The multilevel coordinate search algorithm (MCS) is presented by Huyer

and Neumaier [1999]. The proposed algorithm has powerful theoretical convergence properties. However, for unconstrained problems in four dimensional space or above, the performance of MCS is less satisfactory than for small problems. Huyer and Neumaier compare MCS with existing approaches on different test problems. We make use of their work to compare our proposed method with other approaches.

In a recent survey, Törn *et. al.* [Törn et al., 1999] study the characteristics of unconstrained global optimization test problems. They classify the problems as *unimodal*, *easy*, *moderately difficult* and *difficult* problems. The features considered in this classification are the chance of missing the region of attraction of the global minimum, embeddedness of the global minimum, and the number of local minimizers. An example of a representative set of test problems is also discussed. In our numerical work, we test our algorithm with some of the problems from their paper.

Following our joint report [Birbil et al., 1999], Özdamar and Demirhan developed several heuristic strategies for solving 77 small to moderate size (up to 10 variables) nonlinear test functions and 18 large size test functions (up to 400 variables) collected from literature [2000]. In Chapter 5, we compare a variant of our proposed method with one of the most successful approaches that Özdamar and Demirhan reported.

To test the extended version of our method, we selected several constrained problems from the book by Floudas and Pardalos [1990] who collected a large set of test problems arising in the literature and in a wide range of applications [1990]. These test problems are available at a web site [Floudas et al., 2002].

Neumaier has developed an excellent web site for global optimization [2002]. Like many researchers, we have reached to other useful links through his site. In particular, the test functions used in Chapter 3 and Chapter 4 are collected through these links.

Chapter 3

Electromagnetism-like Mechanism (EM)

In stochastic global optimization, population-based algorithms start with randomly sampling points from the feasible region. According to the objective function values of these sample points, the regions of attraction are determined. Then a mechanism is invoked for further exploitation of these candidate regions. In GAs this mechanism corresponds to the reproduction, crossover and mutation operators [Michalewicz, 1994], whereas in two phase methods the exploration of the feasible region is conducted by random sampling followed by hill-climbing [Rinooy Kan and Timmer, 1987a, Törn and Viitanen, 1994] or gradient-based methods [Fletcher and Reeves, 1964].

Similarly, we construct a mechanism that encourages the points to converge to the highly attractive valleys, and contrarily, discourages the points to move further away from steeper hills. This idea enables us to make an analogy with the attraction-repulsion mechanism of the electromagnetism theory [Birbil and Fang, 2000a,b].

Under this analogy, we can think of each sample point as a charged particle that is released to a space. In our approach, the *charge* of each point relates to the objective function value, which we are trying to optimize. This charge also determines the magnitude

of attraction or repulsion of the point over the sample population - the better the objective function value, the higher the magnitude of attraction.

After calculating these charges, we use them to find a direction for each point to move in subsequent iterations. We select this direction by evaluating a combination force exerted on the point via other points. Like the electromagnetic forces, this force is calculated by adding vectorially the forces from each of the other points calculated separately.

We need to state that though the analogy with electromagnetism theory motivates the idea, there are some notable differences that we make clear when we introduce the method in the subsequent sections.

Finally, similar to the hybrid population-based algorithms [Hart, 1994, Glover et al., 1995], we may apply a local search procedure to improve some of the objective function values observed in the population. In the rest of this chapter, we introduce this new method and present the algorithmic structure. This is followed by the presentation of the numerical results.

3.1 General Scheme for EM

We deal with the problems of the form (1.1) with S defined as in (1.4). Thus, we assume that the following are known from the problem:

- n : dimension of the problem.
- u_k : upper bound in the k^{th} dimension.
- l_k : lower bound in the k^{th} dimension.
- $f(x)$: pointer to the function that is minimized.

The proposed method EM consists of four phases: *initialization* of the sample points, calculation of the *total force* exerted on each particle ¹, *movement* along the direction of the force, and application of *neighborhood search* to exploit the local minima. The general scheme is given in Algorithm 1. The parameters are explained in the following subsections, when we thoroughly describe each part of the general scheme with accompanying algorithms.

Algorithm 1 EM($m, MAXITER, LSITER, \delta$)

m : number of sample points

$MAXITER$: maximum number of iterations

$LSITER$: maximum number of local search iterations

δ : local search parameter, $\delta \in [0, 1]$

```

1: Initialize()
2: iteration  $\leftarrow$  1
3: while iteration < MAXITER do
4:   Local(LSITER,  $\delta$ )
5:    $\mathbf{F} \leftarrow$  CalcF()
6:   Move( $\mathbf{F}$ )
7:   iteration  $\leftarrow$  iteration + 1
8: end while

```

3.1.1 Initialization

The procedure *Initialize* is used to sample m points randomly from the feasible domain, which is an n dimensional hyper-cube. Each coordinate of a point is assumed to be uniformly distributed between the corresponding upper bound and lower bound. After a point is sampled from the feasible region, the objective function value for the point is calculated using the function pointer $f(x)$ (Algorithm 2, line 6). The procedure ends with m points identified, and the point that has the best function value stored in x^{best} (line 8).

¹We use the words *particle* and *point* interchangeably throughout the dissertation.

Algorithm 2 Initialize()

```

1: for  $i = 1$  to  $m$  do
2:   for  $k = 1$  to  $n$  do
3:      $\lambda \leftarrow U(0, 1)$ 
4:      $x_k^i \leftarrow l_k + \lambda(u_k - l_k)$ 
5:   end for
6:   Calculate  $f(x^i)$ 
7: end for
8:  $x^{best} \leftarrow \operatorname{argmin}\{f(x^i), \forall i\}$ 

```

3.1.2 Local Search

The procedure *Local* is used to gather the local information for a point x^i . The parameters, *LSITER* and δ that are passed to this procedure, represent the number of iterations and the multiplier for the neighborhood search, respectively.

The procedure iterates as follows: First, the maximum feasible step length (*Length*) is calculated by using the parameter δ (Algorithm 3, line 2). Second, for a given i , improvement in x^i is sought coordinate by coordinate (lines 5-13). For a given coordinate, the point x^i is assigned to a temporary point y to store the initial information. Next, a random number is selected as a step length and the point y is moved along that direction. If the point y observes a better point within *LSITER* iterations, the point x^i is replaced by y and the neighborhood search for point i ends (lines 14-17). Finally the *current best* point is updated (line 22).

This is a simple random line search algorithm applied coordinate by coordinate. This procedure does not require any gradient information to perform the local search. Instead of using other powerful local search methods, we have utilized this procedure because we wanted to show that even with this trivial method, the algorithm shows promising convergence properties.

Algorithm 3 Local(*LSITER*, δ)

```

1: counter  $\leftarrow$  1
2: Length  $\leftarrow$   $\delta(\max_k\{u_k - l_k\})$ 
3: for  $i = 1$  to  $m$  do
4:   for  $k = 1$  to  $n$  do
5:      $\lambda_1 \leftarrow U(0, 1)$ 
6:     while counter < LSITER do
7:        $y \leftarrow x^i$ 
8:        $\lambda_2 \leftarrow U(0, 1)$ 
9:       if  $\lambda_1 > 0.5$  then
10:         $y_k \leftarrow y_k + \lambda_2(\textit{Length})$ 
11:       else
12:         $y_k \leftarrow y_k - \lambda_2(\textit{Length})$ 
13:       end if
14:       if  $f(y) < f(x^i)$  then
15:         $x^i \leftarrow y$ 
16:        counter  $\leftarrow$  LSITER - 1
17:       end if
18:       counter  $\leftarrow$  counter + 1
19:     end while
20:   end for
21: end for
22:  $x^{best} \leftarrow \operatorname{argmin}\{f(x^i), \forall i\}$ 

```

3.1.3 Calculation of Total Force Vector

The *superposition principle* of electromagnetism theory states that the force exerted on a point via other points is inversely proportional to the distance between the points and directly proportional to the product of their charges [Cowan, 1968].

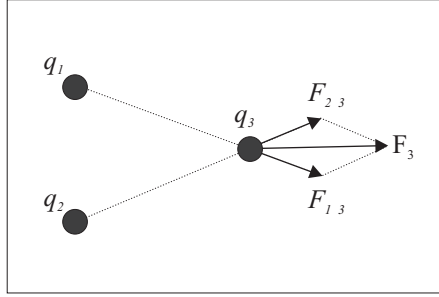


Figure 3.1: Superposition Principle ($\vec{F}_{i3} = \frac{q_3 q_i}{4\pi \epsilon_0 \epsilon_r r^2} \vec{e}_r$ $i = 1, 2$).

Analogously, in each iteration we determine the charge of every point according to the objective function values of the points. However, in our method the charge of each point is not constant and changes from iteration to iteration.

The charge of each point, q^i determines point i 's power of attraction or repulsion. This charge is evaluated by

$$q^i = \exp\left(-n \frac{f(x^i) - f(x^{best})}{\sum_{k=1}^m (f(x^k) - f(x^{best}))}\right), \forall i. \quad (3.1)$$

In this way, points that have better objective values possess higher charges. We multiply the fraction by the dimension n , because in higher dimensions the number of points in the population tends to get large. As a result of this, the fraction may become very small, and may cause overflow problems in calculating the exponential function.

We define the charge, q^i according to the relative efficiency of the objective function

value of the corresponding point in the population. This is clearly not the unique nor the optimal choice for this calculation. An alternative calculation, which rank the points according to their objective function values, may be used here. Our experiments have shown that the proposed calculation in Equation (3.1) is satisfactory for our study.

Notice that, unlike electrical charges, no signs are attached to the charge of an individual point in Equation (3.1). Instead, we decide the direction of a particular force between two points after comparing their objective function values. Hence, the total force F^i exerted on point i is computed by the following:

$$F^i = \sum_{j \neq i}^m \left\{ \begin{array}{ll} (x^j - x^i) \frac{q^i q^j}{\|x^j - x^i\|^2} & \text{if } f(x^j) < f(x^i) \\ (x^i - x^j) \frac{q^i q^j}{\|x^j - x^i\|^2} & \text{if } f(x^j) \geq f(x^i) \end{array} \right\}, \forall i \quad (3.2)$$

Algorithm 4 CalcF():F

```

1: for  $i = 1$  to  $m$  do
2:    $q^i \leftarrow \exp(-n \frac{f(x^i) - f(x^{best})}{\sum_{k=1}^m (f(x^k) - f(x^{best}))})$ 
3:    $F^i \leftarrow 0$ 
4: end for
5: for  $i = 1$  to  $m$  do
6:   for  $j = 1$  to  $m$  do
7:     if  $f(x^j) < f(x^i)$  then
8:        $F^i \leftarrow F^i + (x^j - x^i) \frac{q^i q^j}{\|x^j - x^i\|^2}$  {Attraction}
9:     else
10:       $F^i \leftarrow F^i - (x^j - x^i) \frac{q^i q^j}{\|x^j - x^i\|^2}$  {Repulsion}
11:    end if
12:  end for
13: end for

```

As seen in Algorithm 4 (lines 7-8), between two points, the point that has a better objective function value attracts the other one. Contrarily, the point with worse objective function value repels the other (lines 9-10). Since x^{best} has the minimum objective function value, it acts as an absolute point of attraction, i.e., it attracts all other points in the

population.

When we examine the algorithm carefully, we can see that the determination of a direction via total force vector resembles the statistical estimation of the gradient of f . However, the estimation computed by our method is different since this direction depends on the Euclidean distance between the points. That is, the points that become close enough may lead each other to a direction other than the statistically estimated one.

3.1.4 Movement According to Total Force Vector

After evaluating the total force vector F^i , the point i is moved in the direction of the force by a random step length as given in Equation (3.3). Here the random step length, λ , is assumed to be uniformly distributed between 0 and 1. Obviously, there are many other distributions that can be used in calculation of this step length. But for ease of computation, we have applied uniform distribution. We have selected the step length randomly in order to ensure that the points have a nonzero probability to move to the unvisited regions along this direction.

In Equation (3.3), RNG is a vector whose components denote the allowed feasible movement toward the upper bound, u^k , or the lower bound, l^k , for the corresponding dimension (Algorithm 5, lines 6-10). Furthermore, the force exerted on each particle is normalized so that we can maintain the feasibility. Thus,

$$x^i = x^i + \lambda \frac{F^i}{\|F^i\|} (RNG) \quad i = 1, 2, \dots, m. \quad (3.3)$$

Algorithm 5 gives the *Move* procedure. Note that the best point, x^{best} , is not moved and is carried to the subsequent iterations (line 2). This suggests that we may avoid the calculation of the total force vector on the current best point in Algorithm 4 (yet the computational

effort for calculating the total force vector on a single point is negligible).

Algorithm 5 Move(**F**)

```

1: for  $i = 1$  to  $m$  do
2:   if  $i \neq best$  then
3:      $\lambda \leftarrow U(0, 1)$ 
4:      $F^i \leftarrow \frac{F^i}{\|F^i\|}$ 
5:     for  $k = 1$  to  $n$  do
6:       if  $F_k^i > 0$  then
7:          $x_k^i \leftarrow x_k^i + \lambda F_k^i (u_k - x_k^i)$ 
8:       else
9:          $x_k^i \leftarrow x_k^i + \lambda F_k^i (x_k^i - l_k)$ 
10:      end if
11:    end for
12:  end if
13: end for

```

3.1.5 Termination Criteria

In our study we terminate EM by using a maximum number of iterations. According to our test results, in general 25 iterations per dimension (i.e., $MAXITER = 25n$) is satisfactory for solving the moderate difficulty functions.

Another termination criterion that might be used is the successive number of iterations spent without changing the current best point. In other words, if the current best point is not changed for certain number of iterations, the algorithm may be stopped. However this decision has to be studied carefully since algorithm may be terminated before converging to the global optimum. On the other hand, unnecessary function evaluations may be avoided by stopping earlier.

In the literature several other stopping conditions are studied [Boender and Romeijn, 1995, Törn et al., 1999]. One of the frequently used criteria is to terminate the algorithm

when the observed objective function value is ε -close to the optimal value [Huyer and Neumaier, 1999]. However, this criterion is not appropriate if the global optimum is not known in advance.

3.2 Computational Results

In a recent paper, Törn et al. classified the well-known global optimization problems as *unimodal*, *easy*, *moderately difficult*, and *difficult* problems [1999]. They also suggested that the test problems should represent different classes. We have included some of the functions that Törn et al. discussed and we have created a test function set consisting of 15 functions. We refer to this test function set as the *general test functions*. The functions and their references are given in Appendix B.1.

Initially, we studied three versions of EM, which differ in the local search procedure. We demonstrate our results for the general test functions in terms of the average number of function evaluations over 25 runs. The average and best objective function values are also reported. After selecting the best version, we applied EM to a well known test set, which consists of seven test functions [Dixon and Szegö, 1975]. In addition to reporting our results, we compare the results with other methods reported by Huyer and Neumaier [1999].

All the computations were conducted on a Pentium-III 450 MHz PC. The algorithm has been coded in C++ and it is available upon request.

3.2.1 General Test Functions

We tested EM with three different versions of the *Local* procedure. First, we excluded the local procedure completely from the general scheme. Second, we applied the procedure to all sample points. Finally, we utilized the local search procedure with the best point only.

The input parameters for the functions are given in Table 3.1. The functions are presented in alphabetical order, so the order of a function does not necessarily represent the difficulty of the corresponding function.

Table 3.1: Parameters for General Test Functions

Function Name	n	m	MAXITER	LSITER	δ
Complex	2	10	50	10	5.0e-3
Davis	2	20	50	30	5.0e-3
Epistacity(4)	4	30	50	10	1.0e-3
Epistacity(5)	5	40	100	20	1.0e-3
Griewank	2	30	100	20	1.0e-3
Himmelblau	2	10	50	5	1.0e-3
Kearfott	4	10	50	5	1.0e-3
Levy	10	20	75	5	1.0e-3
Rastrigin	2	20	50	10	5.0e-3
Sine Envelope	2	20	75	10	5.0e-4
Stenger	2	10	75	10	1.0e-3
Step	5	10	50	5	1.0e-3
Spiky	2	30	75	10	1.0e-3
Trid(5)	5	10	125	50	1.0e-2
Trid(20)	20	40	500	150	1.0e-3

EM without Local procedure

In order to examine the basic convergence properties of EM, we first excluded the *Local* procedure from the algorithm. This was achieved simply by setting the parameter LSITER

Table 3.2: Results of EM without *Local* procedure

Function Name	Avg. Evals.	Avg $f(x)$	Best $f(x)$	Known Optimum
Complex	363	0.0175	0.0158	0.0
Davis	622	1.6157	1.5641	0.0
Epistacity(4)	1079	0.0379	0.0149	0.0
Epistacity(5)	2603	0.0355	0.0207	0.0
Griewank	1914	0.0896	0.0032	0.0
Himmelblau	84	0.0934	0.0759	0.0
Kearfott	231	0.0008	0.0000	0.0
Levy	835	0.1429	0.0303	0.0
Rastrigin	141	-1.9566	-1.9871	-2.0
Sine Envelope	962	0.0744	0.0400	0.0
Stenger	282	0.0020	0.0019	0.0
Step	728	0.0000	0.0000	0.0
Spiky	1702	-38.6378	-38.7251	-38.85
Trid(5)	968	-28.2997	-29.0324	-30.0
Trid(20)	43354	-33.2567	-177.6124	-1520.0

to 0. The immediate effect of this choice is the loss of the local information. In this case, even when a point is close to a local optimizer, it is not directed deeper into the valley. In certain sense, this approach behaves blindly. However, since no additional information is collected, the average number of function evaluation figures are not large.

The results in Table 3.2 show that the solutions for the functions *Davis* and *Trid(20)* are less satisfactory than those for the other functions. *Davis* is highly irregular in the neighborhood of the optimum, while the feasible region of *Trid(20)* is a 20-dimensional hypercube with bounds $[-400, 400]$ in each dimension. Therefore the number of evaluations in an experiment is not large enough to adequately explore the feasible space.

Our results show that even if we do not use the *Local* procedure, the average function values are still good. In most of the problems, EM is able to approximate the optimum.

Table 3.3: Results of EM with *Local* procedure applied to all points

Function Name	Avg. Evals.	Avg $f(x)$	Best $f(x)$	Known Optimum
Complex	5534	0.0000	0.0000	0.0
Davis	8091	0.4088	0.1322	0.0
Epistacity(4)	32823	0.0003	0.0001	0.0
Epistacity(5)	189769	0.0001	0.0001	0.0
Griewank	50790	0.0000	0.0000	0.0
Himmelblau	3287	0.0000	0.0000	0.0
Kearfott	6190	0.0000	0.0000	0.0
Levy	44814	0.0001	0.0000	0.0
Rastrigin	10359	-2.0000	-2.0000	-2.0
Sine Envelope	15629	0.0116	0.0001	0.0
Stenger	8316	0.0000	0.0000	0.0
Step	5743	0.0000	0.0000	0.0
Spiky	10264	-38.8004	-38.8492	-38.85
Trid(5)	37154	-29.9979	-29.9999	-30.0
Trid(20)	1.5e+6	-1519.6117	-1519.9768	-1520.0

Nevertheless, the average objective function value figures show that in some cases, EM did not converge to the global optimum, and the accuracy of the objective function values degraded. This motivated the next experiment.

EM with Local procedure applied to all points

We next applied the *Local* procedure to all points in the population. This approach has two advantages; first, the attractive parts of the feasible region are more thoroughly examined, and second, the repelled points have better chance to lead to as yet undiscovered minimizers.

Notice that the performance of the algorithm improves but at the cost of the number of function evaluations required by the *Local* procedure (Table 3.3). Especially, the accuracy

Table 3.4: Results of EM with *Local* procedure applied to current best point

Function Name	Avg. Evals.	Avg $f(x)$	Best $f(x)$	Known Optimum
Complex	598	0.0000	0.0000	0.0
Davis	832	0.4538	0.2356	0.0
Epistacity(4)	1580	0.0002	0.0001	0.0
Epistacity(5)	4123	0.0002	0.0000	0.0
Griewank	2470	0.0000	0.0000	0.0
Himmelblau	520	0.0001	0.0000	0.0
Kearfott	712	0.0000	0.0000	0.0
Levy	2783	0.0001	0.0000	0.0
Rastrigin	792	-1.9898	-2.0000	-2.0
Sine Envelope	1007	0.0352	0.0097	0.0
Stenger	724	0.0000	0.0000	0.0
Step	870	0.0000	0.0000	0.0
Spiky	1520	-38.6684	-38.8486	-38.85
Trid(5)	1870	-29.9963	-29.9997	-30.0
Trid(20)	99731	-1519.4472	-1519.5543	-1520.0

of the results for the functions *Davis*, *Levy*, and *Trid(20)* are much better. However, the number of evaluations drastically increases for the problems with high dimensionality.

EM with Local procedure applied to the current best point only

To reduce the number of function evaluations, we tried applying *Local* procedure to the current best point only (Table 3.4). This choice balances the number of evaluations and the accuracy of the results.

In this case, the average number of evaluations is closer to those of the first version (without *Local* procedure), while the quality of the results is comparable to those of the second version (*Local* procedure applied to all points). However, when the function has good attractors spread over the feasible region as in functions *Sine Envelope* and *Spiky*,

Table 3.5: Results of EM for Dixon and Szegö [1975] test functions

Function [(a)]	n	m	MAXITER	Avg. Evals.	Avg $f(x)$	Best $f(x)$	f_{glob}
Shekel [S5] (b)	4	40	150	3368	-9.7320	-10.1532	-10.1532
Shekel [S7]	4	40	150	1782	-10.4024	-10.4029	-10.4029
Shekel [S10]	4	40	150	5620	-10.5109	-10.5109	-10.5364
Hartman [H3]	3	30	75	1114	-3.8625	-3.8628	-3.8628
Hartman [H6]	6	30	75	2341	-3.3072	-3.3224	-3.3224
Goldstein Price [GP]	2	20	50	420	3.0001	3.0000	3.0000
Branin [BR]	2	20	50	315	0.3980	0.3979	0.3979
Six Hump Camel [C6]	2	20	50	233	-1.0316	-1.0316	-1.0316
Shubert [SHU]	2	20	50	358	-186.7227	-186.7309	-186.7309

(a) : The labels of the functions are given in brackets [Huyer and Neumaier, 1999].

(b) : 2 of the 25 runs do not converge to the global optimum.

the points visiting the neighborhood of the global optimum may not be powerful enough to attract other points. In function *Davis*, the algorithm rapidly converges to the neighborhood of the optimum point, but the region around global optimum has many local minimizers in steep valleys. Thus, EM is trapped in one of these local minima.

3.2.2 Dixon and Szegö Test Functions

Our initial experiments show that EM rapidly converges to the minimizers. The local information gains more importance either when the function is highly irregular in the neighborhood of the global optimum, or when the global optimum is far from the highly attractive local minimizers. Although the application of the local search to all points provides the best results, the improvement over applying local search only to the current best point is not significant. Therefore, for most of the problems, applying local search to all points may turn out to be overkill. In the subsequent experimentation, we utilized the third version of EM for comparison with other global optimization methods.

For the comparison, we used the standard test functions given by Dixon and Szegö

[1975]. The performance of different methods on these functions has been extensively studied by Huyer and Neumaier [1999]. We use their results in comparing EM with the existing approaches. As a stopping criterion we use the following Equation (3.4), that is also applied by Huyer and Neumaier [1999]:

$$\frac{(f(x^{best}) - f_{glob})}{|f_{glob}|} \leq 10^{-4} \quad (3.4)$$

where f_{glob} represents the known global optimum.

Table 3.5 shows our results with the corresponding parameters. EM does not exhibit any difficulty to converge to the global optimum in all functions, except *Shekel* [S5]. This function has an attractive local minimizer, which is far from the global optimum.

Table 3.6 shows the comparison of EM with other methods in terms of number of function evaluations. Except for the last row, the figures in the table are taken from [Huyer and Neumaier, 1999]. From the table we see that MCS achieves the best results followed generally speaking by the other methods (labeled with (f) and (h)), which use first or second order information in the neighborhood search. EM does as well as or better than the most of the older methods indicated in the upper part of the table.

Overall, our additional comments are in order:

- Unlike MCS and other methods indicated with (f), EM does not use the first or the second order information.
- EM is able to provide answers for all the problems.
- All the methods, other than EM that are included in Table 3.6, have some limitations due to their rigid structures. EM has a very flexible design which would permit using the desirable features from other methods.
- In both sets of test problems, we have observed that EM performs well when

Table 3.6: Comparison of EM with different methods in terms of number of function evaluations

Method	S5	S7	S10	H3	H6	GP	BR	C6	SHU
Bremmerman	(a)	(a)	(a)	(a)	(a)	(a)	250		
Mod. Bremmerman	(a)	(a)	(a)	(a)	515	300	160		
Zilinskas	(a)	(a)	(a)	8641			5129		
Gomulka-Branin	5500	5020	4860						
Törn	3679	3606	3874	2584	3447	2499	1558		
Gomulka-Törn	6654	6084	6144						
Gomulka-V.M.	7085	6684	7352	6766	11125	1495	1318		
Price	3800	4900	4400	2400	7600	2500	1800		
Faggioli	2514	2519	2518	513	2916	158	1600		
De Biase-Frontini	620	788	1160	732	807	378	587		
Mockus	1174	1279	1209	513	1232	362	189		
Bélisle et al. (b)				339	302	4728	1846		
Boender et al. (f)	567	624	755	235	462	398	235		
Snyman-Fatti (f)	845	799	920	365	517	474		178	
Kostrowicki-Piela (g)	(c)	(c)	(c)	200	200	120		120	
Yao (f)								1132	≤ 6000
Perttunen (f)	516	371	250	264		82	97	54	197
Perttunen-Stuckman (f)	109	109	109	140	175	113	109	96	(a)
Jones et al. (h)	155	145	145	199	571	191	195	285	2967
Storn-Price (d)	6400	6194	6251	476	7220	1018	1190	416	1371
MCS (e) (f)	83	129	103	79	111	81	41	42	69
EM	3368	1782	5620	1114	2341	420	315	233	358

(a) Method converged to a local minimum.

(b) Average number of function evaluations when converges; for H6, converged only 70 percent of time.

(c) Global minimum not found within 12000 function calls.

(d) Average over 25 cases. For H6, average over 24 cases only; one case did not converge within 12000 function calls.

(e) The version that gives the best results is selected.

(f) Recent methods that use first or second order information.

(g) Requires closed form for a particular integral.

(h) Partitions the search space into hyper-rectangles.

Missing entry means that no result is available [Huyer and Neumaier, 1999]

there is a pattern in the function. However, if there are multiple attractive local minimizers spread around the feasible region, EM may be deceived and the performance of it may decrease.

- The combination of the general scheme with other powerful local search methods [Solis and Wets, 1981], may increase the efficiency of the algorithm.
- The computational results suggest that we could improve our results by using first or second order information in the local search.

3.3 Effects of Different Parameters

The proposed approach depends on several parameters. In order to analyze the effects of these parameters, we have selected two functions -one from each set- and conducted a general factorial design with three factors.

We have selected the number of sample points (m), the maximum number of iterations ($MAXITER$), and the local search parameter (δ) as our factors (A, B and C respectively). We have not added the maximum number of local search iterations since in the local search procedure the big *while* loop ends as soon as a better neighboring solution is found (Algorithm 3, lines 14-17).

The levels of the factors are given in Table 3.7. We select the levels of the first two factors according to the dimension of the problem (n). In each combination of the levels we have used 10 replicates, thus totally 270 runs have been taken.

We have used the function *Sine Envelope* from the first test set and the function *Shekel* [S5] from the second set. We have selected these functions because the computational results have shown that the efficiency of EM on these functions is not as good as its efficiency

Table 3.7: Levels of the factors used in experimental design

	Factors		
	A (m)	B ($MAXITER$)	C (δ)
Level 1	5n	25n	1.0e-4
Level 2	10n	50n	1.0e-3
Level 3	20n	75n	1.0e-2

on other functions in the corresponding test sets.

Table 3.8 and Table 3.9 show that the significant factors of the model are A, B and AC. These results suggest that when we run the method for a *long time* or when we *increase the number of points* in the population the chance of finding the global optimum increases. Besides, if we change the local search parameter, δ , with the number of points in the population the results improve.

Table 3.8: Analysis of variance table for the function *Sine Envelope*

	Sum of Squares	Degrees of Freedom	Mean Square	F Value
A	5.421e-3	2	2.71e-3	4.11
B	2.2e-2	2	1.1e-2	16.35
C	1.037e-3	2	5.186e-4	0.79
AB	3.474e-3	4	8.684e-4	1.32
AC	6.470e-3	4	1.617e-3	2.45
BC	1.9e-3	4	4.751e-4	0.72
ABC	3.013e-3	8	3.767e-4	0.57
Error	0.16	243	6.596e-4	
Total	0.20	269		

Table 3.9: Analysis of variance table for the function *Shekel* [S5]

	Sum of Squares	Degrees of Freedom	Mean Square	F Value
A	48.58	2	24.29	3.01
B	71.41	2	35.71	4.43
C	42.42	2	21.21	2.63
AB	49.42	4	12.35	1.53
AC	83.16	4	20.79	2.58
BC	30.58	4	7.65	0.95
ABC	126.83	8	15.85	1.97
Error	1958.36	243	8.06	
Total	2410.76	269		

3.4 Summary

In this chapter we have introduced a new approach, EM, which is a powerful yet easy method for global optimization. The proposed method mimics the electromagnetism theory of physics by considering each sample point as an electrical charge. Like most multiple-point stochastic methods, the proposed method starts with sampling random points from the feasible region. Then it proceeds according to the objective function values of the points at the subsequent iterations. The strength of the method lies in the idea of directing the sample points toward local optimizers by utilizing an attraction-repulsion mechanism.

We have applied the algorithm to different test problems and compared our results with other methods from the literature. Without using the first or second order information, EM converges rapidly to optimum when the number of function evaluations is used as a performance measure. We also note that EM can be used as a stand-alone approach or as an accompanying algorithm for other methods.

In the next chapter, a thorough study of the convergence properties of EM will be studied. We will first examine the stochastic process generated by the proposed method and after some modifications, we will analyze the global convergence in probability.

Chapter 4

Theoretical Structure

There are convergence proofs for random search methods in the literature [Baba, 1981, Dekkers and Aarts, 1991, Pintér, 1984, Boender and Romeijn, 1995]. In an early work, Yakowitz and Fisher studied a general framework for the conditions of convergence of stochastic search methods [Yakowitz and Fisher, 1973]. Most of these proofs are based on a critical mechanism, which ensures that the algorithm does not discard any part of the feasible region. Usually, this is accomplished by adding a uniform sampling procedure which has a small probability of being invoked. Although, the effect of this procedure is minimal from a computational stand point, its existence is extremely important for the convergence result. This mechanism works like a *life-saver* to guarantee that no matter how the method behaves, there exists a nonzero probability of covering any full-dimensional subset in S .

In this chapter, we focus on refining the major procedures of the EM so that we can come up with an asymptotic convergence proof to the global optimum for the refined method. Relative to our previous study we introduce a *free particle* to avoid premature convergence and to elevate the strength of the method [Birbil et al., 2001]. The necessary

changes in the method are explained in detail in the following sections.

4.1 Modifications on the Original Method

Before introducing the modifications, we discuss the necessity of the changes by elaborating on “premature convergence”.

4.1.1 Premature Convergence

The premature convergence may occur when the forces exerted on the particles omit some parts of the feasible region. We illustrate this phenomenon by an example.

In Figure 4.1, the function has the optimum at point 0, and it has a high peak that is close to this point. After the peak, the function is monotonically decreasing. If all the particles in the current population were located on the right hand side of the peak, in the original EM, all particles would be directed toward the right, which would end up with a local minimizer, and the algorithm would converge prematurely.

In order to preclude premature convergence, we have to somehow “perturb” the current population so that at least one of the particles will have a chance to move to the possibly omitted parts of the feasible region. Hence, one of the particles in the population other than the current best one will be selected as the *perturbed point* and denoted by x^p . Next, the *CalcF* procedure is modified to take into account this perturbed point.

4.1.2 Refined General Scheme

First, we restate the general scheme in Algorithm 6. Here we assume that the following parameters are given: dimension of the problem (n), upper and lower bounds in the k^{th}

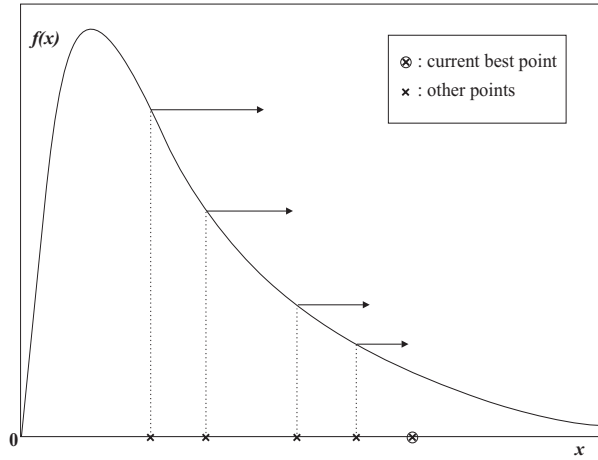


Figure 4.1: An example of premature convergence on one dimensional space

dimension, (u_k, l_k) , and pointer to the function (f). Note that except the total force calculation procedure, the rest of the previous method is preserved.

Recall that m represents the number of points in the population. We have added a new parameter, ν , which we call the *threshold parameter* for use with the free particle. This parameter will be explained in the next section.

Algorithm 6 EM(m, ν)

- 1: Initialize()
 - 2: **while** termination criteria are not satisfied **do**
 - 3: Local()
 - 4: $\mathbf{F} \leftarrow \text{CalcF}(\nu)$
 - 5: Move(\mathbf{F})
 - 6: **end while**
-

As before, we adopt the notation, x^i , to specify the i^{th} point of the population. However, in addition to the current best point, x^{best} , there is another point in the population which is indexed separately. This is the free particle, which we refer to as the *perturbed point*, x^p .

4.1.3 The Perturbed Point and The Modified *CalcF* Procedure

The modified *CalcF* procedure is given in Algorithm 7. Note that a new parameter $\nu \in (0, 1)$ is introduced, which will be described below. The perturbed point, x^p is selected as the farthest point from the current best point, x^{best} in the current population (line 1, Algorithm 7), i.e.,

$$x^p \triangleq \arg \max \{ \|x^{best} - x^i\|, i = 1, 2, \dots, m \}. \quad (4.1)$$

Algorithm 7 CalcF(ν)

```

1:  $x^p \leftarrow \arg \max \{ \|x^{best} - x^i\|, i = 1, 2, \dots, m \}$ 
2: for  $i = 1$  to  $m$  do
3:    $q^i \leftarrow \exp(-n \frac{f(x^i) - f(x^{best})}{\sum_{k=1}^m (f(x^k) - f(x^{best}))})$ 
4:    $F^i \leftarrow 0$ 
5: end for
6: for  $i = 1$  to  $m$  do
7:   for  $j = 1$  to  $m$  do
8:     if  $i \neq j$  then
9:       if  $x^i \neq x^p$  then
10:         $F_j^i \leftarrow (x^j - x^i) \frac{q^i q^j}{\|x^j - x^i\|^2}$ 
11:       else
12:         $\lambda \leftarrow U(0, 1)$ 
13:         $F_j^i \leftarrow (x^j - x^i) \frac{\lambda q^i q^j}{\|x^j - x^i\|^2}$ 
14:        if  $\lambda < \nu$  then
15:           $F_j^i \leftarrow -F_j^i$  {Reverse Direction}
16:        end if
17:       end if
18:       if  $f(x^j) < f(x^i)$  then
19:         $F^i \leftarrow F^i + F_j^i$  {Attraction}
20:       else
21:         $F^i \leftarrow F^i - F_j^i$  {Repulsion}
22:       end if
23:     end if
24:   end for
25: end for

```

Except x^p , the calculation of the total force vector remains the same for all points (see Equation (3.2)). For x^p , the component forces are perturbed by a random number λ , i.e.,

$$F_j^p = \left\{ \begin{array}{ll} (x^j - x^p) \frac{\lambda q^p q^j}{\|x^j - x^p\|^2}, & \text{if } f(x^j) < f(x^p) \\ (x^p - x^j) \frac{\lambda q^p q^j}{\|x^j - x^p\|^2}, & \text{if } f(x^p) \leq f(x^j) \end{array} \right\} \quad (4.2)$$

where λ is uniformly distributed between 0 and 1 (lines 12-13, Algorithm 7). Also, the directions of the component forces are perturbed; that is, if the random variable λ is less than the parameter ν then the direction of the component force is reversed (Algorithm 7, lines 14-16). Consequently, there exists one point in the population for which the direction of movement may be reversed.

4.2 Convergence Results for the Modified EM

The steps of the modified EM is given in Algorithm 8. Notice that the neighborhood search procedure *Local* is not included in this revised version, since it does not effect the convergence proof. Our task is to show that in the long run (i.e., when $MAXITER \rightarrow \infty$) the modified method converges to the set of global optima with probability one. In the following sections, we give the convergence proof after introducing the notation.

Algorithm 8 EM(m, ν)

- 1: Initialize()
 - 2: iteration \leftarrow 1
 - 3: **while** iteration \leq MAXITER **do**
 - 4: CalcF(ν)
 - 5: Move()
 - 6: iteration \leftarrow iteration + 1
 - 7: **end while**
-

4.2.1 Notation and Assumptions

We first recall the definition of the global optimum solution: Let $f : \mathfrak{R}^n \rightarrow \mathfrak{R}$ and $S \subset \mathfrak{R}^n$ be as in (1.1), then $x^* \in \mathfrak{R}^n$ is called a *global minimizer* on S , if

$$x^* \in S \text{ and } f(x^*) \leq f(x), \forall x \in S. \quad (4.3)$$

This leads to the definition of the set of points that are in the vicinity of global minimizer. Given $\varepsilon > 0$, the *set of ε -optimal solutions* is defined by

$$B_\varepsilon^* = \{x \in S : |f(x) - f(x^*)| \leq \varepsilon\}. \quad (4.4)$$

In our derivation we make the following assumptions:

1. B_ε^* contains an open ball of full dimensionality, i.e., $\mu(B_\varepsilon^*) > 0$, where μ is the Lebesgue measure on \mathfrak{R}^n .
2. $f : S \rightarrow \mathfrak{R}$ is a lower bounded measurable function with respect to the Lebesgue measure μ .
3. The collection of vectors (corresponding to the m ($m \geq n + 1$) points in the current population) at every iteration generated by the algorithm has full rank, i.e.,

$$\text{rank}(\{x^1, x^2, \dots, x^m\}) = n. \quad (4.5)$$

Recall that the proposed method (Algorithm 8) utilizes a stochastic search mechanism with a population of points. Thus, there exists an underlying stochastic process, which depends on the location of the m points in the feasible region S .

Formally, if we define Y_k as the random variable corresponding to the collection of m vectors at iteration k , then the stochastic process generated by the algorithm becomes the

family of random variables $\{Y_k; k = 0, 1, 2, \dots\}$. Also, the collection of these m vectors corresponds to the *state* of the process. Hence, we define the *state space* as

$$\mathbb{X}_m \triangleq \{\mathbf{x} : \mathbf{x} = (x^1, x^2, \dots, x^m), x^i \in S, i = 1, 2, \dots, m\}, \quad (4.6)$$

where \mathbf{x} denotes a state. In this setting, the random variable Y_k gives the state of the process at iteration k .

Notice that, in the algorithm the location of the points at the next iteration depends only on the current population. Therefore, the stochastic process generated by the algorithm constitutes a time homogeneous Markov Chain [Taylor and Karlin, 1998]. This leads to the definition of the transition probability $\rho(\mathbf{x}, A)$; for any given $\mathbf{x} \in \mathbb{X}_m$, and $A \subset S$

$$\rho(\mathbf{x}, A) \triangleq P\{\chi_A(Y_{k+1}) \neq 0 \mid Y_k = \mathbf{x}\} \quad (4.7)$$

evaluates the conditional probability of making a transition from state \mathbf{x} into a state that has *at least* one point in A , where

$$\chi_A(\mathbf{x}) \triangleq \sum_{i=1}^m \mathbf{1}_A(x^i) \quad (4.8)$$

gives the number of points in $A \cap \mathbf{x}$. $\mathbf{1}_A$ is the indicator function for set A , i.e., $\mathbf{1}_A(x^i)$ returns 1 if x^i is in A and returns 0 otherwise.

Remember that the main modification on the original method is the addition of the perturbed point, x^p . This point plays an essential role in the proof of convergence to the global optimum, so we introduce additional notation related to x^p . Let $d \in \mathfrak{R}^n$ be any direction, then

$$L(x^p, d) \triangleq \{x(\eta) : x(\eta) = x^p + \eta d \in \mathfrak{R}^n, \eta \geq 0\} \quad (4.9)$$

denotes the *ray* originated at x^p and directed along d . Furthermore, for any given subset A

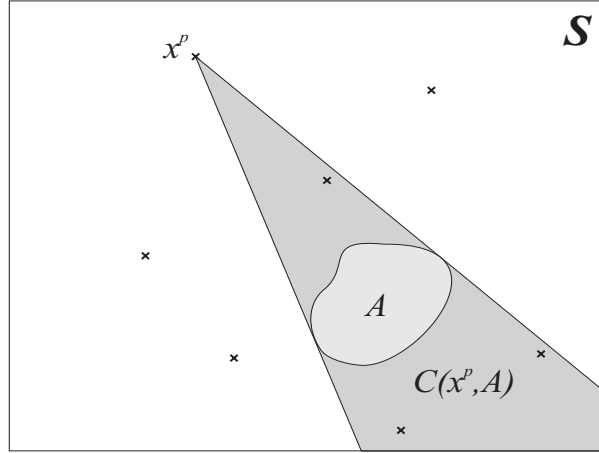


Figure 4.2: Truncated cone in \mathbb{R}^2 .

of S , we define

$$C(x^p, A) \triangleq \{x : x \in L(x^p, y - x^p) \cap S, \text{ for some } y \in A\} \quad (4.10)$$

as the *truncated cone* pointed at x^p (Figure 4.2).

4.2.2 Convergence With Probability One

In this section we present a proof of convergence to the global optimum. First, let us define the concept of convergence for the modified method. Given $\varepsilon > 0$, if there exists an integer $K(\varepsilon) \geq 0$ such that

$$\chi_{B_\varepsilon}(Y_k) \neq 0, \forall k > K(\varepsilon), \quad (4.11)$$

then the modified EM (Algorithm 8) is said to *converge* to the set of ε -optimal solutions.

Recall the definition of $\rho(\mathbf{x}, A)$ in (4.7), we further define

$$\rho^*(A) \triangleq \inf_{\mathbf{x} \in \mathbb{X}_m} \{\rho(\mathbf{x}, A)\} \quad (4.12)$$

for a given subset A of S .

Lemma 1 *Given $A \subset S$, if A contains an open ball of full dimensionality in S , then*

$$\rho^*(A) > 0. \quad (4.13)$$

Proof. Let $\mathbf{x} \in \mathbb{X}_m$ be a population of points at any particular iteration. Notice that by assumption 3, this collection of vectors has full rank. Without loss of generality we may assume that $m = n + 1$, so when we focus on the perturbed point x^p , the component forces F_j^p become a basis. Also, in Algorithm 7 (lines 14-16), the directions of the component forces may be reversed with a positive probability. Therefore, any direction around x^p can be possibly generated as a combination of the component forces. Furthermore, two more steps are needed so that x^p may move into the given set A at the next iteration. First, a direction vector, which falls into the truncated cone $C(x^p, A)$ should be generated (Figure 4.2). Then, an appropriate step length is required so that x^p can be displaced into A .

Next we show that there exist nonzero lower bounds on the probabilities of the steps stated above. Multiplying the values of the lower bounds for each step provides a positive lower bound for $\rho^*(A)$.

Step 1. Since A contains an open ball with full dimensionality in S , there exists a ball $B_r \subset A$ with its radius $r > 0$ being small enough such that B_r lies in the nonnegative combination of the component forces at x^p with the directions of some of the component forces being reversed. Let $\kappa_1(\mathbf{x})$ denote the probability for reversing the directions of some of the component forces so that B_r lies in the nonnegative combination of the component forces. Note that in Algorithm 7 (line 14) the probability of reversing each component force is $\nu \in (0, 1)$. Therefore, the lower bound on the probability of

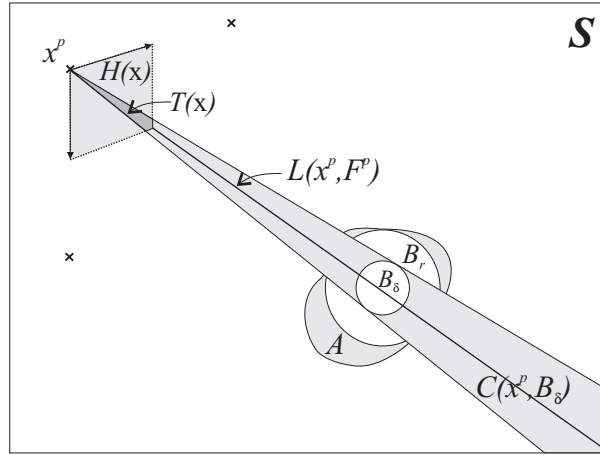


Figure 4.3: Calculation of the probability for Step 2 in \mathfrak{R}^2 .

Step 1 is

$$0 < \kappa_1^* \triangleq \min\{(1 - \nu)^n, \nu^n\} \leq \kappa_1(\mathbf{x}). \quad (4.14)$$

Step 2. Let B_δ be a closed ball with radius $\delta > 0$ in B_r , i.e.,

$$B_\delta \subset B_r \subset A. \quad (4.15)$$

The proper selection of B_δ will become apparent shortly.

By assumption 3, the component forces, F_j^p , (with some directions being reversed) form a basis. Let $H(\mathbf{x})$ be the set generated by the component force vectors, i.e.,

$$H(\mathbf{x}) \triangleq \left\{ x : x = \sum_{j \neq p}^m \gamma_j F_j^p, \gamma_j \in [0, 1] \right\}. \quad (4.16)$$

Equation (4.2) shows that the component forces are perturbed with uniformly distributed numbers between 0 and 1. As illustrated in Figure 4.3, let

$$T(\mathbf{x}) \triangleq \mu(H(\mathbf{x}) \cap C(x^p, B_\delta)) \quad (4.17)$$

then the probability of generating a total force vector that lies in $C(x^p, B_\delta)$ is

$$\kappa_2(\mathbf{x}) \triangleq \frac{\mu(T(\mathbf{x}))}{\mu(H(\mathbf{x}) \cap S)}. \quad (4.18)$$

The value of $\kappa_2(\mathbf{x})$ decreases as the denominator increases and the numerator decreases. The upper bound of the denominator is the volume of S , i.e.,

$$\mu(H(\mathbf{x}) \cap S) \leq \prod_{k=1}^n (u_k - l_k). \quad (4.19)$$

In (4.18), the volume of $T(\mathbf{x})$ depends on the volume of the $H(\mathbf{x})$, and the distance between B_δ and x^p . Let,

$$\alpha \triangleq \inf_{x \in B_\delta} \{\|x - x^p\|\}. \quad (4.20)$$

be the distance between B_δ and x^p . Since S is bounded, we have

$$\alpha \leq \alpha^* \triangleq \sup_{x, y \in S} \{\|x - y\|\}. \quad (4.21)$$

Also note that by (4.16), the volume of $H(\mathbf{x})$ decreases as the lengths of the component force vectors decrease. Let us define,

$$\beta \triangleq \min_{j \neq p} \|F_j^p\|. \quad (4.22)$$

By Equation (4.2) we have

$$\|F_j^p\| = \frac{q^p q^j}{\|x^j - x^p\|}, \quad (4.23)$$

where the charges of the points are evaluated as in Equation (3.1). Note that in (3.1), the fraction in the exponential function is between 0 and 1. Hence, the lower bound on the charge of any point is $q^* \triangleq e^{-n}$. By using (4.21), we have

$$0 < \beta^* \triangleq \frac{(q^*)^2}{\alpha^*} \leq \beta. \quad (4.24)$$

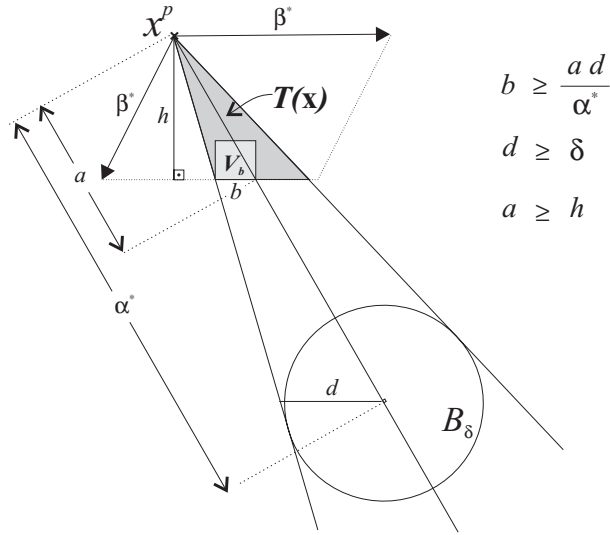


Figure 4.4: Calculation of the lower bound for $\mu(T(\mathbf{x}))$ in \mathfrak{R}^2 .

Let $V_b \subset T(\mathbf{x})$ be a cube with a side length of b . As illustrated in Figure 4.4, such cube exists since we can select δ arbitrarily small. Note that this result can be further generalized to the n dimensional case as

$$b^n = \mu(V_b) \leq \mu(T(\mathbf{x})). \quad (4.25)$$

Next we show that b is bounded away from 0. From Figure 4.4, we see $a \geq h$, $d \geq \delta$ and

$$b \geq \frac{ad}{\alpha^*}. \quad (4.26)$$

With δ sufficiently small, the Pythagoras theorem implies

$$h^2 + (\beta^* - b)^2 \geq (\beta^*)^2 \quad (4.27)$$

$$a \geq h \geq \sqrt{2b\beta^* - b^2} \quad (4.28)$$

Substituting the lower bounds of a and d into (4.26) gives

$$b \geq \frac{\delta \sqrt{2b\beta^* - b^2}}{\alpha^*} \quad (4.29)$$

Hence,

$$b^2 \geq \frac{\delta^2(2\beta^*b - b^2)}{(\alpha^*)^2} \quad (4.30)$$

$$(\alpha^*)^2 b^2 \geq 2\beta^* b \delta^2 - b^2 \delta^2 \quad (4.31)$$

$$b((\alpha^*)^2 + \delta^2) \geq 2\beta^* \delta^2 \quad (4.32)$$

$$b \geq \frac{2\beta^* \delta^2}{(\alpha^*)^2 + \delta^2}. \quad (4.33)$$

Consequently, the lower bound on b becomes

$$b^* \triangleq \frac{2\beta^* \delta^2}{(\alpha^*)^2 + \delta^2}. \quad (4.34)$$

This further leads to

$$0 < (b^*)^n \leq \mu(V_b) \leq \mu(T(\mathbf{x})). \quad (4.35)$$

Therefore, the lower bound on the probability of Step 2 is

$$0 < \kappa_2^* \triangleq \frac{(b^*)^n}{\prod_{k=1}^n (u_k - l_k)} \leq \kappa_2(\mathbf{x}). \quad (4.36)$$

Step 3. Let $L(x^p, F^p)$ be the ray originated at x^p and directed along F^p (Figure 4.3). In Algorithm 5 the perturbed point is moved along the total force vector by a uniformly distributed step length (between 0 and up/down to the boundaries). Hence, let

$$L(x^p, F^p) \cap \partial B_r \triangleq \{y_1, y_2\} \text{ and } L(x^p, F^p) \cap \partial S \triangleq \{z\}, \quad (4.37)$$

where ∂B_r and ∂S denote the boundaries of the B_r and S , respectively. Then, the probability of moving into A is

$$\kappa_3(\mathbf{x}) \triangleq \frac{\|y_1 - y_2\|}{\|x^p - z\|}. \quad (4.38)$$

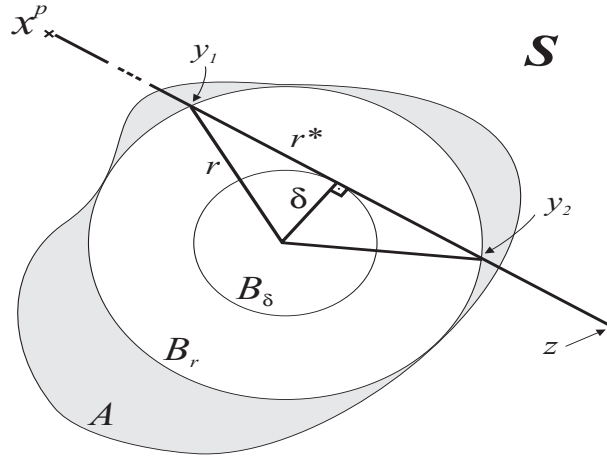


Figure 4.5: The calculation of r^* in \mathbb{R}^2 .

As illustrated in Figure 4.5, if we define

$$r^* \triangleq \sqrt{r^2 - \delta^2}, \quad (4.39)$$

then

$$0 < 2r^* \leq \|y_1 - y_2\|. \quad (4.40)$$

Therefore, by using (4.21) the lower bound on the probability of Step 3 is

$$0 < \kappa_3^* \triangleq \frac{2r^*}{\alpha^*} \leq \kappa_3(\mathbf{x}). \quad (4.41)$$

Therefore we have shown that

$$0 < \kappa_1^* \kappa_2^* \kappa_3^* \leq \rho^*(A). \quad (4.42)$$

This completes the proof of the lemma. \square

Assumption 1 ensures that B_ε^* contains an open ball of full dimensionality. Therefore, there exists a nonzero probability for the population at any given iteration to move into the set of ε -optimal solutions in one iteration.

The next lemma shows that at any iteration, if one of the points is in B_ε^* then at subsequent iterations there always exists at least one point residing in B_ε^* . Intuitively, this reflects an *absorbing event* for the algorithm.

Lemma 2 *Given any state $\mathbf{x} \in \mathbb{X}_m$ and $k \geq 0$ if $\chi_{B_\varepsilon^*}(Y_k) \neq 0$ then $P\{\chi_{B_\varepsilon^*}(Y_{k+1}) \neq 0 \mid Y_k = \mathbf{x}\} = 1$.*

Proof. Suppose at iteration k ,

$$Y_k = \mathbf{x} \text{ and } \chi_{B_\varepsilon^*}(Y_k) \neq 0, \quad (4.43)$$

then we know that $x^{best} \in B_\varepsilon^*$. Algorithm 5 (line 2) ensures that unless another point observes a better objective function value than that of x^{best} , the current best point in B_ε^* remains at iteration $k + 1$. If not, then x^{best} is replaced by this new point, which again resides in B_ε^* . Thus,

$$\chi_{B_\varepsilon^*}(Y_{k+1}) \neq 0. \quad (4.44)$$

This completes the proof of the lemma. \square

We are now ready to prove that the modified EM converges to the set of ε -optimal solutions with probability one.

Theorem 1 *Provided that the assumptions in Section 4.1 hold, Algorithm 8 converges to B_ε^* with probability one, i.e.,*

$$\lim_{k \uparrow \infty} P\{\chi_{B_\varepsilon^*}(Y_k) \neq 0\} = 1. \quad (4.45)$$

Proof. The Markovian property of the stochastic process and Lemma 2 imply that

$$\begin{aligned} P\{\chi_{B_\varepsilon^*}(Y_k) = 0\} &= P\{\chi_{B_\varepsilon^*}(Y_1) = 0, \chi_{B_\varepsilon^*}(Y_2) = 0, \dots, \chi_{B_\varepsilon^*}(Y_k) = 0\} \\ &= P\{\chi_{B_\varepsilon^*}(Y_1) = 0\} \prod_{l=2}^k P\{\chi_{B_\varepsilon^*}(Y_l) = 0 \mid \chi_{B_\varepsilon^*}(Y_{l-1}) = 0\} \end{aligned} \quad (4.46)$$

Since we have a time homogeneous Markov Chain, it is sufficient to compute only

$$P\{\chi_{B_\varepsilon^*}(Y_l) = 0 \mid \chi_{B_\varepsilon^*}(Y_{l-1}) = 0\}. \quad (4.47)$$

Let us define

$$D = \{\mathbf{x} : \chi_{B_\varepsilon^*}(\mathbf{x}) = 0\}, \quad (4.48)$$

then we have,

$$\begin{aligned} P\{\chi_{B_\varepsilon^*}(Y_l) = 0 \mid \chi_{B_\varepsilon^*}(Y_{l-1}) = 0\} &= \frac{P\{\chi_{B_\varepsilon^*}(Y_{l-1})=0, \chi_{B_\varepsilon^*}(Y_{l-1})=0\}}{P\{\chi_{B_\varepsilon^*}(Y_l)=0\}} \\ &= \frac{\int_D P\{\chi_{B_\varepsilon^*}(Y_l)=0 \mid Y_l=\mathbf{y}\} P\{Y_{l-1}=\mathbf{y}\} \mu(d\mathbf{y})}{\int_D P\{Y_{l-1}=\mathbf{y}\} \mu(d\mathbf{y})} \end{aligned} \quad (4.49)$$

Following (4.12), let $\rho^*(B_\varepsilon^*) = \rho^*$, and by Lemma 1, we have

$$P\{\chi_{B_\varepsilon^*}(Y_l) = 0 \mid Y_{l-1} = \mathbf{y}\} \leq (1 - \rho^*), \quad \forall \mathbf{y} \in D. \quad (4.50)$$

Hence (4.49) becomes

$$\begin{aligned} P\{\chi_{B_\varepsilon^*}(Y_l) = 0 \mid \chi_{B_\varepsilon^*}(Y_{l-1}) = 0\} &\leq \frac{(1-\rho^*) \int_D P\{Y_{l-1}=\mathbf{y}\} \mu(d\mathbf{y})}{\int_D P\{Y_{l-1}=\mathbf{y}\} \mu(d\mathbf{y})} \\ &= (1 - \rho^*). \end{aligned} \quad (4.51)$$

Substituting (4.51) into (4.46), we get

$$P\{\chi_{B_\varepsilon^*}(Y_k) = 0\} \leq (1 - \rho^*)^k. \quad (4.52)$$

By Lemma 2, we have

$$\begin{aligned} \lim_{k \uparrow \infty} P\{\chi_{B_\varepsilon^*}(Y_k) \neq 0\} &= 1 - \lim_{k \uparrow \infty} P\{\chi_{B_\varepsilon^*}(Y_k) = 0\} \\ &\geq 1 - \lim_{k \uparrow \infty} (1 - \rho^*)^k = 1. \end{aligned} \quad (4.53)$$

This completes the proof of the theorem. \square

4.2.3 Computational Considerations

The main theorem shows that Algorithm 8 eventually reaches an ε -optimal solution. If we consider the output of the algorithm in each iteration as a sequence of Bernoulli trials, then the average number of failures before the first success can be calculated [Law and Kelton, 2000]. By Lemma 1, the probability of success is $\rho^*(B_\varepsilon^*)$. Therefore the average number of failures before the first success is $(1 - \rho^*(B_\varepsilon^*)) / \rho^*(B_\varepsilon^*)$.

Again for computational study, we need to discuss two possible overflow problems. First, recall that the charge of each point is calculated by (3.1). Hence, if the objective function attains very high values, the fraction may become too small and cause an overflow problem in calculating the exponential function. This problem can be avoided by assigning a large floating point value (depending of the word length of the particular computer) to the points with very high objective function values. Second, if the distance between two points is close to zero, then there may be an overflow problem due to the denominator of the fractions in equations (3.2) and (4.2). This can be also avoided by setting a small enough number to the distance between these points according to the word length of the computer.

4.3 Computational Results

We tested the refined method (Algorithm 6) with two sets of functions. First we solved the same collection of problems Dixon and Szegö [1975] test functions. However since this set is regarded in the literature as easy, we selected hard problems from the field (which include problems of higher dimension) to form an additional test set.

As before, the initialization procedure is the uniform sampling from the feasible region. We applied the previous local search method (Algorithm 3) to the current best point only.

We did not adjust the parameters $LSITER$ and δ used in the *Local* procedure and set them to be 10 and $1.0e-3$, respectively. In our experiments, we used maximum number of iterations (MAXITER), and in all runs the threshold parameter (ν) was set to be 0.25. As the stopping criterion, we again used equation (3.4).

The following two sections show the results for each function. The tables of the results include the average and best objective function values, as well as the average number of function evaluations.

4.3.1 Dixon and Szegö Test Functions

Table 4.1 shows that the refined algorithm does not have any difficulty in solving the problems in this test set. Also, the number of function evaluations required to converge to the optimum is still satisfactory. When we compare the results with the results of the previous version of the algorithm (Table 3.5), we can see that for *Shekel* functions [S5], [S7], [S10], and for functions *Hartman* [H6] and *Shubert* [SHU], the refined version gives a smaller average number of function evaluations. On the other hand for the functions *Hartman* [H3], *Branin* [BR] and *Six Hump Camel* [C6] the average number of function evaluations is slightly larger.

The observed increase in the number of the function evaluations for the latter functions is due to the perturbed point. This is what we would expect from the refined version. However, the decrease for the other functions was unexpected. This suggests that the addition of the perturbed point aids in exploration of the unvisited parts of the feasible domain.

Table 4.1: Results for Dixon and Szegö [1975] test functions with the refined algorithm

Function	n	m	MAXITER	Avg. Evals.	Avg $f(x)$	Best $f(x)$	f_{glob}
Shekel [S5]	4	40	150	2800	-9.54637	-10.1532	-10.1532
Shekel [S7]	4	40	150	1608	-10.4024	-10.4029	-10.4029
Shekel [S10]	4	40	150	5445	-10.5109	-10.5109	-10.5364
Hartman [H3]	3	30	75	1303	-3.8626	-3.8628	-3.8628
Hartman [H6]	6	30	75	2206	-3.3045	-3.3224	-3.3224
Goldstein Price [GP]	2	20	50	421	3.0001	3.0000	3.0000
Branin [BR]	2	20	50	393	0.3979	0.3979	0.3979
Six Hump Camel [C6]	2	20	50	253	-1.0316	-1.0316	-1.0316
Shubert [SHU]	2	20	50	265	-185.1975	-186.7309	-186.7309

4.3.2 Hard Test Functions

We selected 4 functions, among which the dimension (n) varies between 4 and 64. These problems are suggested to test the ability of a global optimization method to reach the global minimizer and to discriminate it from other local minimizers [Neumaier, 2002].

Table 4.2 shows the results of the modified method with hard test functions. The first two functions $Perm(4, 0.005)$ and $Perm0(10, 100)$ are very hard problems since they contain many local minima, which lead to large differences in function values in case of small discrepancies in the variable values. For these two functions, EM is able to converge to the global optimum within 25 runs. However the average objective function values show that EM gets trapped in one of the local minimizers and does not converge to the global minimizer.

Powersum (8) and Powersum (64) are two test functions that have singular minimizers among very flat valleys. Therefore, the Hessians are ill-conditioned at the solution and the problems become very hard for methods that use higher order information. Moreover, for Powersum (64) the dimension is very large and can not be efficiently handled by many methods. As the average objective function values point out, EM performs extremely well

for these two functions.

The number of function evaluation figures for all functions look promising. Furthermore, this number can be drastically decreased by using a more accurate and rapidly convergent local search procedure.

Table 4.2: Results for hard test functions

Function	n	m	MAXITER	Avg. Evals.	Avg $f(x)$	Best $f(x)$	f_{glob}
Perm(4, 0.005)	4	20	150	5181	0.2541	0.0033	0.0
Perm0(10, 100)	10	50	250	32718	0.9438	0.0133	0.0
Powersum (8)	8	40	200	5646	0.0001	0.0000	0.0
Powersum (64)	64	100	500	154678	0.0002	0.0001	0.0

4.4 Summary

This chapter gives a proof of convergence to the global optimum for the modified version of the proposed method. Our main task has been to show that when the number of iterations is *large enough*, one of the points in the current population moves into the ε -neighborhood of the global optimum. In order to achieve this result, we have given a detailed mathematical construction, which could be easily applied to some of the other population-based stochastic search methods.

Chapter 5

Variants of EM and Extensions

A number of interesting questions were raised by the computational results of the previous chapters: What is the effect of using higher order information in EM? How can we speed up local convergence for convex minimization? Can we decrease the step lengths gradually so that the search process could be intensified in particular regions? In this chapter, we tackle these questions by developing particular variants of EM. We note that there are many alternatives to answer any one of the questions; the variants presented in this chapter are just one of these many alternatives. In addition, in this chapter we provide a preliminary extension of the EM algorithm for solving constrained optimization problems.

Note that in the following sections, unless we introduce a new variant of EM, we use the modified scheme (Algorithm 6) discussed in the previous chapter.

5.1 Using Higher Order Information

Our experiments in Chapter 3 show that the bulk of the computational effort is due to the *Local* procedure. Though this procedure has served the purpose of showing the effectiveness of EM for locating regions of attraction, it limits the quality of the results in terms of precision and number of function evaluations. Therefore, replacing *Local* procedure by an alternative method that is effective in local refinement is advantageous. Furthermore, an alternative method utilizing first or second order information enables us to make a fair comparison between EM and other methods mentioned in Table 3.6.

In order to easily merge EM with an effective local search algorithm, we used MATLAB. MATLAB is a high-performance language for technical computing that is used extensively in both academia and industry [MathWorks, 2000]. MATLAB provides several algorithms for unconstrained optimization. We selected the procedure *fminunc*, which converges to the *local minimum* of a real-valued function of several variables, starting at an initial estimate. This procedure uses the BFGS Quasi-Newton method with a mixed quadratic and cubic line search procedure. In Algorithm 6, we remove the *Local* procedure, and instead pass the current best point to procedure *fminunc* at each iteration as the initial estimate. In the subsequent experiments, the default parameters provided by MATLAB were used and each call to a test function by the procedure *fminunc* was counted as an additional function evaluation (including the calls for the calculation of the gradient).

Table 5.1 shows the performance of EM using higher order information on the Dixon and Szegö [1975] test functions. For each function, the average number of function evaluations, average objective function values, and the best objective function values from 25 runs are presented. The parameters n , m and MAXITER are the same as in Table 3.5, and equation (3.4) was used as the stopping criterion.

Table 5.1: Results of EM using higher order information for Dixon and Szegö [1975] test functions

Function [(a)]	n	m	MAXITER	Avg. Evals.	Avg. $f(x)$	Best $f(x)$	f_{glob}
Shekel [S5] (b)	4	40	150	221	-9.9511	-10.1532	-10.1532
Shekel [S7]	4	40	150	402	-10.4029	-10.4029	-10.4029
Shekel [S10]	4	40	150	558	-10.5109	-10.5109	-10.5364
Hartman [H3]	3	30	75	99	-3.8628	-3.8628	-3.8628
Hartman [H6]	6	30	75	155	-3.3224	-3.3224	-3.3224
Goldstein Price [GP]	2	20	50	76	3.0000	3.0000	3.0000
Branin [BR]	2	20	50	60	0.3979	0.3979	0.3979
Six Hump Camel [C6]	2	20	50	74	-1.0316	-1.0316	-1.0316
Shubert [SHU]	2	20	50	210	-186.7309	-186.7309	-186.7309

Notice that EM using higher order information is able to converge to the global optimum in all the problems. As the average objective function values column (column 6) indicates, except for the problem *Shekel* [S5], EM was able to locate the global minimizer in all 25 runs. For the problem *Shekel* [S5], algorithm converged to another attractive local minimizer in two of the twenty five runs. Furthermore, in comparison with Table 3.6, the number of function evaluations has decreased dramatically for all the problems.

To show the performance improvement in terms of number of function evaluations, we compare our results with other methods as before. Note that Table 5.2 is Table 3.6 from Chapter 2 with one more row corresponding to EM with higher order information added. From the table we see that EM with higher order information outperforms most of the methods. Moreover, for problem *Goldstein Price* [GP], EM achieves an even better result than the most successful method, MCS.

Table 5.2: Comparison of EM using higher order information with other methods for Dixon and Szegö [1975] test functions

Method	S5	S7	S10	H3	H6	GP	BR	C6	SHU
Bremmerman	(a)	(a)	(a)	(a)	(a)	(a)	250		
Mod. Bremmerman	(a)	(a)	(a)	(a)	515	300	160		
Zilinskas	(a)	(a)	(a)	8641			5129		
Gomulka-Branin	5500	5020	4860						
Törn	3679	3606	3874	2584	3447	2499	1558		
Gomulka-Törn	6654	6084	6144						
Gomulka-V.M.	7085	6684	7352	6766	11125	1495	1318		
Price	3800	4900	4400	2400	7600	2500	1800		
Fagiuoli	2514	2519	2518	513	2916	158	1600		
De Biase-Frontini	620	788	1160	732	807	378	587		
Mockus	1174	1279	1209	513	1232	362	189		
Bélisle et al. (b)				339	302	4728	1846		
Boender et al. (f)	567	624	755	235	462	398	235		
Snyman-Fatti (f)	845	799	920	365	517	474		178	
Kostrowicki-Piela (g)	(c)	(c)	(c)	200	200	120		120	
Yao (f)								1132	≤ 6000
Perttunen (f)	516	371	250	264		82	97	54	197
Perttunen-Stuckman (f)	109	109	109	140	175	113	109	96	(a)
Jones et al. (h)	155	145	145	199	571	191	195	285	2967
Storn-Price (d)	6400	6194	6251	476	7220	1018	1190	416	1371
MCS (e) (f)	83	129	103	79	111	81	41	42	69
EM (i)	3368	1782	5620	1114	2341	420	315	233	358
EM (j)	221	402	558	99	155	76	60	74	210

(a) Method converged to a local minimum.

(b) Average number of function evaluations when converges;
for H6, converged only 70 percent of time.

(c) Global minimum not found within 12000 function calls.

(d) Average over 25 cases. For H6, average over 24 cases only;
one case did not converge within 12000 function calls.

(e) The version that gives the best results is selected.

(f) Recent methods that use first or second order information.

(g) Requires closed form for a particular integral.

(h) Partitions the search space into hyper-rectangles.

(i) Previous scheme used in Chapter 2 (see Table (cite)).

(j) EM using higher order information.

Missing entry means that no result is available [Huyer and Neumaier, 1999]

5.2 EM for Nonsmooth Convex Minimization: EMNCM

The class of *smooth convex optimization functions* had been extensively studied and both theoretically and computationally flourished [Luenberger, 1973, Bertsekas, 1995]. However, the problems in which the objective function belongs to the class of *nonsmooth convex functions*, still constitute a challenging research area in the optimization field [Lavery, 2000].

Although EM has been designed for nonconvex optimization, this section investigates the applicability of EM algorithm to the class of nonsmooth convex minimization problems. Some important problems which are convex but nonsmooth. Next we discuss the modifications in the algorithm and then present some numerical results.

5.2.1 Some Simplifications

To accelerate the performance for (nonsmooth) convex minimization problems, we simplify EM, and denote this new variant by EMNCM. The changes in the refined scheme (Algorithm 6) are summarized as follows:

- The number of points in the population (m) is equal to the dimension of the problem plus one ($n + 1$).
- Since the objective function is convex, every local minimum gives the global optimum. Therefore, we do not need to utilize a perturbed point to prevent premature convergence. So, the threshold parameter ν is set to be 0.
- *Local* procedure is removed from the general scheme (Algorithm 6, line 3), so

that the computational burden of extensive function evaluations in this procedure is avoided ¹.

Furthermore, we change procedure *CalcF* so that all the points are attracted to the other points that have better objective function values than theirs. Only the current best point is repelled by all other points for possible acceleration of the search behavior down toward the valley. Nevertheless, since the repulsion of the current best point may degrade its objective function value, the best objective function value found up to current iteration is stored. New *CalcF* and *Move* procedures after implementing these changes are given in Algorithm 9 and Algorithm 10, respectively. Notice that, in Algorithm 10 the current best point is also moved according to the total force vector.

Algorithm 9 CalcF():F

```

1: for  $i = 1$  to  $m$  do
2:    $q^i \leftarrow \exp\left(-n \frac{f(x^i) - f(x^{best})}{\sum_{k=1}^m (f(x^k) - f(x^{best}))}\right)$ 
3:    $F^i \leftarrow 0$ 
4: end for
5: for  $i = 1$  to  $m$  do
6:   for  $j = 1$  to  $m$  do
7:     if  $f(x^j) < f(x^i)$  then
8:        $F^i \leftarrow F^i + (x^j - x^i) \frac{q^i q^j}{\|x^j - x^i\|^2} \{Attraction\}$ 
9:     else
10:      if  $x^i = x^{best}$  then
11:         $F^i \leftarrow F^i - (x^j - x^i) \frac{q^i q^j}{\|x^j - x^i\|^2} \{Only\ x^{best}\ is\ repelled.\}$ 
12:      end if
13:    end if
14:  end for
15: end for

```

¹If higher precision is required, any line search algorithm (e.g., golden section search) can be applied *after* running EM. However, these algorithms are expensive in terms of function evaluations

Algorithm 10 Move(**F**)

```

1: for  $i = 1$  to  $m$  do
2:    $\lambda \leftarrow U(0, 1)$ 
3:    $F^i \leftarrow \frac{F^i}{\|F^i\|}$ 
4:   for  $k = 1$  to  $n$  do
5:     if  $F_k^i > 0$  then
6:        $x_k^i \leftarrow x_k^i + \lambda F_k^i (u_k - x_k^i)$ 
7:     else
8:        $x_k^i \leftarrow x_k^i + \lambda F_k^i (x_k^i - l_k)$ 
9:     end if
10:  end for
11: end for

```

5.2.2 Comparison with Downhill Simplex Method

In order to test EMNCM, we compare our solutions with the solutions of downhill simplex method on a set of test problems. Our empirical study with EMNCM showed that its behavior was similar to that of the downhill simplex method [Nelder and Mead, 1965], which has enjoyed considerable popularity in the field. Downhill simplex method works with a set of points from the feasible region, where the number of points in the set is one more than the dimension of the problem. Like EMNCM, downhill simplex method does not require first or second order information. We have coded downhill simplex method following Press et al. [1993]. We note that there are other methods specialized in nonsmooth convex minimization and their performances in terms of number of function evaluations may be better than the downhill simplex method [Brent, 1973, Foulds, 1981, Nesterov and Nemirovski, 1994]. However, we selected downhill simplex method since this method and EMNCM have many conceptual similarities.

For this reason, we selected four functions that are being used in an ongoing research project in our department. These functions are derived from the class of shape preserving

cubic L_1 splines that are particularly useful in modeling terrain, geophysical features, biological objects, and financial processes [Lavery, 2000]. The general form of these functions are given in Appendix B.4.

We made 25 runs for each problem. The average number of function evaluations and the average objective function values in 25 runs are shown in Table 5.3. Since the performance of both methods depends on the initial starting population, both algorithms were initialized with the same set of points. This was accomplished by using the same random seed for each run. In the downhill simplex method the tolerance parameter, which shows the distance of the objective function value from the optimum value, is selected to be $1.0e - 6$.

Table 5.3: Comparison of EMNCM and downhill simplex method

Function	EMNCM		Down Hill Simplex		Known Opt.
	Avg. Evals	Avg. $f(x)$	Avg. Evals	Avg. $f(x)$	
F1 ($n = 10$)	762	33.5170	1120	33.7278	33.0000
F2 ($n = 10$)	729	33.1543	1246	32.3426	31.3642
F3 ($n = 13$)	1342	25.8457	1588	25.0679	24.1801
F4 ($n = 13$)	622	619.5965	752	619.6244	619.4210

The average number of function evaluation figures in Table 5.3 shows that EMNCM converges faster than Downhill Simplex Method. However, in functions $F2$ and $F3$ the precision of the results with EMNCM is not as good as Downhill Simplex Method. This is not surprising because Downhill Simplex Method utilizes *contraction steps* to increase the precision. On the other hand, in EMNCM the points are moved up to (down to) the boundaries and there are no precision improvement steps.

5.3 EM with Adaptive Step Length: EMASL

In this section, the effect of an adaptive step length is investigated. Previously in the *Move* procedure, the points were allowed to move up to the boundaries during the iterations. However, this causes EM to slow down, particularly with problems that have large feasible regions. To remedy this, we introduce a minor change in Algorithm 5 to decrease the step length gradually. The new method is denoted by EMASL. After explaining the proposed changes, we compare the new method with a simulated annealing heuristic.

5.3.1 Change In Move Procedure

We replace Equation (3.3) by

$$x^i = x^i + T\lambda \frac{F^i}{\|F^i\|} (RNG) \quad i = 1, 2, \dots, m \quad (5.1)$$

where $T \in [0, 1]$. Initially t is equal to 1, but after each iteration it is reduced as follows:

$$T \leftarrow \frac{T}{1 + \beta T} \quad (5.2)$$

where $\beta \in (0, 1)$ is a constant. Clearly, there exist other formulations for reducing the step length. The motivation behind selecting Equation (5.2) is two-fold: First, this equation is very simple and computationally cheap to apply. Second, Equation (5.2) has been shown to have promising performance with simulated annealing heuristics [Ingber, 1994]. The new *Move* procedure is given in Algorithm 11.

5.3.2 Comparison with a Simulated Annealing Variant

Özdamar and Demirhan have developed several probabilistic search techniques for global optimization [2000]. In their paper, one of the more successful heuristics is a simulated

Algorithm 11 Move(F)

```

1: for  $i = 1$  to  $m$  do
2:    $\lambda \leftarrow U(0, 1)$ 
3:    $F^i \leftarrow \frac{F^i}{\|F^i\|}$ 
4:   for  $k = 1$  to  $n$  do
5:     if  $F_k^i > 0$  then
6:        $x_k^i \leftarrow x_k^i + T\lambda F_k^i(u_k - x_k^i)$ 
7:     else
8:        $x_k^i \leftarrow x_k^i + T\lambda F_k^i(x_k^i - l_k)$ 
9:     end if
10:  end for
11: end for
12:  $T \leftarrow \frac{T}{1+\beta T}$ 

```

annealing variant, in which the neighbour to a current solution is generated randomly as follows. A dimension is selected randomly and the decision is made randomly whether the coordinate of the variable in the selected dimension is to be decreased or increased. According to this decision, a random amount is added/subtracted from the current variable value without violating the corresponding upper and lower bound while the values of the remaining coordinates are unchanged. The new functional value is calculated. The new solution is accepted for sure if the move improves on the current solution. Else, the move is accepted according to the following cooling scheme

$$PA(\Delta f, T) = e^{-\frac{\Delta f}{f_c T}} \quad (5.3)$$

where, PA is the probability of acceptance, f_c is the functional evaluation of the current solution, Δf is the difference between the functional evaluations of the new and the current solutions, and T is the temperature. If a randomly generated number between 0 and 1 turns out to be less than PA , then the not improving move is carried out. T depends on the number of times a not improving move has been obtained consecutively. Initially T is

equal to 1, but after each non-improving move, it is reduced as in Equation (5.2).

We first used the general test functions (B.1) to compare EMASL with the simulated annealing heuristic above (Table 5.4). In this comparison we set β equal to 0.1, which was also used by Özdamar and Demirhan, [2000]. The parameters for EM are the same as before (Table 3.1). As a stopping criterion we applied equation (3.4) for both EM and SA. In case of nonconvergence, both algorithms were stopped after maximum number of iterations was exceeded. Since SA is a single point random search algorithm, for fair comparison maximum number of iterations for SA was set to be $m \times MAXITER$.

Table 5.4 shows that EMASL outperforms SA in terms of number of function evaluations. Except for the problems *Sine Envelope* and *Spiky*, the average objective function values found by EMASL are better than those found with SA. EMASL is able to solve higher dimensional problems (*Levy* and *Trid(20)*) for which SA is not able to converge.

Next, we solved Dixon and Szegö test functions (B.1). Table 5.5 shows that EMASL produces better solutions than SA for all functions. Furthermore, the average number of function evaluation figures suggest that, in general, EMASL is able to converge to the solution in about one tenth of the time that is needed by SA (when SA converges).

5.4 Constrained Optimization Problems

In nonlinear programming, the study of constrained optimization problems plays an important role. In a constrained nonlinear programming problem, the feasible set is defined by a set of equality and/or inequality constraints. A typical constrained programming problem

Table 5.4: Comparison of EMASL with SA for general test functions

Function Name	EM-ASL			SA			f_{glob}
	A.E.	f_{avg}	f_{best}	A.E.	f_{avg}	f_{best}	
Complex	147	0.0000	0.0000	12505	0.0000	0.0000	0.0
Davis	902	0.3557	0.0712	13959	0.9189	0.1281	0.0
Epistacity (4)	1604	0.0002	0.0000	7301	0.0517	0.0193	0.0
Epsitacity (5)	4110	0.0001	0.0000	37946	0.0831	0.0532	0.0
Griewank	1812	0.0000	0.0000	26687	0.0000	0.0000	0.0
Himmelblau	358	0.0000	0.0000	1350	0.0124	0.0000	0.0
Kearfott	764	0.0023	0.0000	1384	0.1598	0.0010	0.0
Levy	2292	0.0000	0.0000				0.0
Rastrigin	516	-1.9953	-1.9999	4621	-1.9978	-1.9999	-2.0
Sine Envelope	1133	0.0316	0.0097	6886	0.0137	0.0097	0.0
Stenger	567	0.0000	0.0000	4295	0.0006	0.0000	0.0
Step	82	0.0000	0.0000	890	0.0000	0.0000	0.0
Spiky	2055	-38.5106	-38.8486	10041	-38.8091	-38.8500	-38.85
Trid(5)	1767	-29.9959	-29.9981	29704	-29.4906	-29.7317	-30.0
Trid(20)	101699	-1519.4484	-1519.5510				-1520.0

A.E.: Average function evaluations.

f_{avg} : Average objective function value in 25 runs.

f_{best} : Best objective function value in 25 runs.

f_{glob} : Known optimum from the literature.

takes the form

$$\begin{aligned} & \text{minimize} && f(x) \\ & \text{subject to} && x \in S \end{aligned} \tag{5.4}$$

where

$$S = \{x \in \mathfrak{R}^n : h_i(x) = 0, i = 1, 2, \dots, k, \quad g_j(x) \leq 0, j = 1, 2, \dots, l\} \tag{5.5}$$

and $f : \mathfrak{R}^n \rightarrow \mathfrak{R}$, $h_i : \mathfrak{R}^n \rightarrow \mathfrak{R}$, $g_j : \mathfrak{R}^n \rightarrow \mathfrak{R}$ are real-valued functions.

Commonly used methods to handle the constraints can be classified into the following three classes: Methods in the first class try to solve (5.4) by applying the idea of iterative descent within the confines of the constraint set. In other words, the search for the optimal solutions starts from a feasible point and at subsequent iterations feasible points are selected

Table 5.5: Comparison of EMASL with SA for Dixon and Szegö [1975] test functions

Function Name	EMASL			SA			f_{glob}
	A.E.	f_{avg}	f_{best}	A.E.	f_{avg}	f_{best}	
Shekel [S5]	1919	-9.3443	-10.1532	59491	-5.4225	-10.1378	-10.1532
Shekel [S7]	1576	-10.4023	-10.4028	69752	-5.4082	-10.3883	-10.4029
Shekel [S10]	5630	-10.5108	-10.5108	67958	-5.5610	-10.5052	-10.5364
Hartman [H3]	942	-3.8626	-3.8627	22404	-3.8617	-3.8626	-3.8628
Hartman [H6]	1702	-3.3222	-3.3223	24343	-3.2385	-3.3037	-3.3224
Goldstein Price [GP]	440	3.0001	3.0000	11303	3.0059	3.0000	3.0000
Branin [BR]	457	0.3979	0.3979	9631	0.3991	0.3979	0.3979
Six Hump Camel [C6]	243	-1.0315	-1.0316	8121	-1.0314	-1.0316	-1.0316
Shubert [SHU]	447	-186.7207	-186.7297	11508	-186.6701	-186.7297	-186.7309

A.E.: Average function evaluations.

f_{avg} : Average objective function value in 25 runs.

f_{best} : Best objective function value in 25 runs.

f_{glob} : Known optimum from the literature.

along the descent direction. The methods in the second class focus on solving a system of equalities and inequalities that correspond to the necessary conditions of optimality [Bertsekas, 1982]. The methods in the last class try to find the optimal solutions of a constrained optimization problem by solving a sequence of unconstrained problems. Here, each unconstrained problem corresponds to an approximation to the constrained optimization problem [Kowalik and Osborne, 1968, Luenberger, 1973].

EM has been specifically designed for solving optimization problems with bound constraints. Through the use of the methods in the third class, constrained optimization problems can be cast into a bound constrained form such that EM can be applied. To explore the possible application of EM to constrained optimization, two frequently used methods were selected. These were the penalty and the barrier methods, both of which offer a direct approach to handle complex constraints [Luenberger, 1973].

5.4.1 Penalty and Barrier Methods

Penalty and barrier methods are based on approximating the constrained optimization problems by unconstrained ones. In both methods, the main idea is to add new cost terms to the objective function that reflect the feasibility or infeasibility of the solution at any iteration. In the case of penalty methods, a high cost associated with the violated constraint is added to the objective function. In the case of barrier methods, the feasible points that are far from the boundaries of the feasible set are favored over the ones that are close to the boundaries [Luenberger, 1973].

Penalty methods transform the constrained problem (5.4) into an unconstrained problem of the form

$$\text{minimize } f(x) + \sum_{i=1}^k c_i \Phi_i(h_i(x)) + \sum_{j=1}^l d_j \Psi_j(g_j(x)) \quad (5.6)$$

where $c_i > 0, i = 1, 2, \dots, k$ and $d_j > 0, j = 1, 2, \dots, l$ are called *penalty coefficients*, and $\Phi_i : \mathbb{R}^n \rightarrow \mathbb{R}, i = 1, 2, \dots, k, \Psi_j : \mathbb{R}^n \rightarrow \mathbb{R}, j = 1, 2, \dots, l$ are called *penalty functions*. Penalty functions with penalty coefficients assign high costs to the points if they violate the constraints.

Barrier methods are similar to penalty methods but unlike penalty methods they are only applicable to problems with inequality constraints. This is because these methods work within the interior of the feasible set and approach the boundaries from the interior. Suppose S consists of only inequalities, then the transformation of problem (5.4) by barrier methods is as follows:

$$\begin{aligned} \text{minimize } & f(x) + \sum_{j=1}^l \frac{1}{d_j} \Gamma_j(g_j(x)) \\ \text{subject to } & x \in \text{int}(S) \end{aligned} \quad (5.7)$$

where Γ_j are barrier functions and $int(s)$ denotes the interior of set S . Barrier functions assign high costs to the points that are close to the boundaries of the feasible region.

It has been shown that as the penalty coefficients go to infinity, the solution points of (5.6) and (5.7) converge to the solution of (5.4) [Luenberger, 1973, Bertsekas, 1982]. Although both penalty and barrier methods are easy to implement, finding appropriate penalty coefficients c_i and d_j for a specific problem instance is not trivial. If the coefficients are too large, constraint satisfaction dominates the search. Furthermore, large penalty terms make the objective function spiky and difficult to search for good solutions. On the other hand, if the penalty coefficients are too small, then the optimal solution reported by either method may not be feasible. Therefore depending on the problem, penalty and barrier methods require the tuning of penalty coefficients either before or during the search process.

5.4.2 Computational Study

In our computational study, all problems were selected to be inequality constrained so that barrier methods can be applied. In addition, we selected particular problems with bounds on the variables. Therefore, the test problems were all of the following form:

$$\begin{aligned} & \text{minimize} && f(x) \\ & \text{subject to} && g_j(x) \leq 0 \quad j = 1, 2, \dots, l \\ & && lb_i \leq x_i \leq ub_i \quad i = 1, 2, \dots, n \end{aligned} \tag{5.8}$$

where $lb_i, ub_i, i = 1, 2, \dots, n$ correspond to the lower and upper bounds, respectively.

For each constraint $g_j(x) \leq 0$, we used the following penalty and barrier functions:

$$\Psi_j(g_j(x)) = [\max(0, g_j(x))]^2 \tag{5.9}$$

$$\Gamma_j(g_j(x)) = -\frac{1}{g_j(x)}. \tag{5.10}$$

Our aim at this point is to present some preliminary results. Therefore, we did not concentrate on tuning the coefficients. To simplify the burden of selecting an appropriate coefficient for each constraint, we selected a scalar value $d > 0$ and set $d_1 = d_2 = \dots = d_l = d$.

Thus, corresponding to the problem (5.8), we considered the following two approximate problems:

$$\begin{aligned} \text{minimize} \quad & f(x) + d \sum_{j=1}^l [\max(0, g_j(x))]^2 \\ & lb_i \leq x_i \leq ub_i \quad i = 1, 2, \dots, n \end{aligned} \quad (5.11)$$

and

$$\begin{aligned} \text{minimize} \quad & f(x) - \frac{1}{d} \sum_{j=1}^l \frac{1}{g_j(x)} \\ & lb_i \leq x_i \leq ub_i \quad i = 1, 2, \dots, n \\ & x \in \text{int}(S') \end{aligned} \quad (5.12)$$

where $S' = \{x \in \mathfrak{R}^n : g_j(x) \leq 0, j = 1, 2, \dots, l\}$. Notice that we did not assign costs to the bound constraints, because these constraints are never violated by EM.

We tried to select the problems that feature interesting properties such as nonconvex functions and disconnected regions. The first problem consists of a nonconvex quadratic objective function, subject to nonconvex quadratic inequality constraints. The second problem also contains nonconvex quadratic inequalities with a concave objective function. The objective function in the third problem is a very simple linear function: however, the feasible region consists of two disconnected subregions. The last two problems include both nonconvex constraints and nonconvex functions. They fall into the category of Geometric Programming. Hence efficient solution procedures exist for both problems. The problems are explicitly given in Appendix B.5.

The parameters for the test problems are given in Table 5.6. We used the same parameters in both penalty and barrier methods for comparison. The last column of the table shows

Table 5.6: Parameters for constrained test problems

Name	n	m	MAXITER	LSITER	δ	ν	d
TP 1	5	30	75	10	0.01	0.25	1.0e+5
TP 2	6	40	100	10	0.01	0.25	1.0e+5
TP 3	2	20	50	10	0.01	0.25	1.0e+3
TP 4	3	30	50	10	0.01	0.25	1.0e+5
TP 5	4	20	750	10	0.01	0.25	1.0e+4

the penalty coefficient d used for the corresponding test problem. We selected the smallest possible penalty coefficient that ensures feasibility. In other words, we started with a small value of the penalty coefficient and increased it until the solution reported by the EM was feasible. After setting the penalty coefficients, 10 runs were made for each problem.

Table 5.7: Comparison of penalty and barrier methods on constrained test problems

Test Problem	Penalty Method			Barrier Method			f_{glob}
	A.E.	f_{avg}	f_{best}	A.E.	f_{avg}	f_{best}	
TP1	2534	-30374.4670	-30563.2846	2264	-30447.6823	-30596.7844	-30665.5387
TP2	4311	-293.9035	-297.7095	3969	-297.8254	-307.2018	-310.0000
TP3	886	-5.4256	-5.5036	885	-5.4245	-5.4756	-5.5079
TP4	1597	-82.5450	-83.096	1347	-81.9877	-83.1574	-83.2540
TP5	1092	-5.4857	-5.6450	1251	-5.0523	-5.6346	-5.7398

A.E.: Average function evaluations.

f_{avg} : Average objective function value in 10 runs.

f_{best} : Best objective function value in 10 runs.

f_{glob} : Known optimum from the literature.

Table 5.7 shows the results of applying EM with penalty and barrier methods. Except for test problem *TP5*, the barrier method outperforms the penalty method in terms of number of function evaluations. When we look at the average objective function values, penalty method is more successful than the barrier method for test problems *TP3*, *TP4* and *TP5*. However, for problem *TP4* barrier method's best objective function value is closer to the

global optimum solution than the penalty method's best objective function value.

5.5 Summary

Among many possible alternatives for extending EM, we have selected several important variations and studied them in this chapter. First, we have explored merging EM with a local optimization procedure that utilizes higher order information. For this purpose, we have used a variant of Quasi Newton method provided by the MATLAB software and replaced the simple local search procedure of the previous chapters with it. This new scheme has been tested on a set of problems and has been compared with other methods. The resulting improvement in the performance of EM was significant. EM was able to outperform most of the other methods of the literature.

Second, we have studied the effect of gradually decreasing the step length. For this purpose, we have adapted a *cooling scheme* from a Simulated Annealing (SA) variant. We compared the modified EM with an SA scheme that has been shown to have a very good performance. Not only did EM outperform SA in terms of function evaluations, it was also able to provide solutions to the problems for which SA failed.

Third, EM algorithm has been modified so that it can be applied to convex minimization. In particular, we have studied the performance of EM for nonsmooth convex minimization problems. After modifying EM, we have compared EM with the downhill simplex method on a set of nonsmooth convex minimization problems. Though EM converged more rapidly than the downhill simplex method, in two of the four test problems downhill simplex method provided a better solution.

Finally, we have conducted a preliminary study on the application of EM to constrained

optimization problems. We have used penalty and barrier methods to transform the constrained optimization problems into unconstrained problems. As a test vehicle, we have prepared a set of problems that belong to different classes of constrained optimization problems. EM performed well on all the problems. However, EM was not able to precisely locate the optimum solution, because of the penalty coefficients. Also, our experiments have shown that EM is able to find initial feasible solutions quickly.

Chapter 6

Conclusion and Further Research

In this chapter, we summarize the work we have done and point out some directions for future research.

6.1 Overview of Accomplishments

We have studied a novel population-based global optimization method, which we call *Electromagnetism-like Mechanism* (EM). The method imitates the behavior of electrically charged particles; a set of points in the population corresponding to a set of charged particles. The strength of the method lies in the idea of directing sample points toward local optimizers, which point out attractive regions of the feasible space.

We have applied the proposed method to different test problems from the literature and compared our results with those of other reported methods. Without using the higher order information, EM has converged rapidly (in terms of the number of function evaluations) to the global optimum and produced highly efficient results for problems of varying degree of difficulty.

We have given a proof of convergence to the global optimum for the proposed method. In order to achieve this result, we refined the original method and provided a systematic way for studying the underlying stochastic process. The thrust of our proof has been to show that in the limit, at least one of the points in the population moves to the neighborhood of the global optimum with probability one.

The structure of the proposed method is very flexible permitting the easy development of variations. Capitalizing on this, we have developed several variants of EM and compared these variants with the other methods from the literature. These variants of EM have been able to provide accurate answers to selected problems and in many cases have been able to outperform other well-known methods.

While with the work to date we have accomplished many of our objectives, we have also formed some other interesting ideas for future research.

6.2 Future Research Directions

The potential research directions may be summarized as follows:

- In EM, the total force vector provides a direction for a point to move at the next iteration. In the case of differentiable functions, another direction can be found by computing the gradient of the function at this particular point. The direction provided by EM is global in the sense that the points are directed toward attractive regions anywhere in the feasible domain. On the other hand, the direction computed via gradient provides a local information that moves the points toward the local minimizers. Total force vector helps in overcoming

the local minimizers, however it may miss the deep valleys in the neighborhood of the point. Therefore, the directions provided by the total force vector and the gradient may be combined. This combination of the local and global behavior of the function may be useful for developing effective algorithms that are capable of avoiding entrapment at local minimizers.

- In neural networks, one of the crucial parts of the learning process is to solve a multimodal nonlinear function [Rumelhart and McClelland, 1986]. Many local optimization methods, such as gradient descent methods and conjugate gradient methods, have been used. These methods guarantee to find only local minimizers. To overcome the local minimizers some deterministic global optimization methods, such as branch and bound methods and trajectory methods, have been applied. However, these methods can only solve small scale networks. Some stochastic global optimization methods and heuristic strategies, such as pure random search, simulated annealing and genetic algorithms, have been also proposed. However, these methods require significantly large number of iterations to solve the problems. As we showed through our experiments, EM can overcome local minimizers as well as it does not require large number of iterations. Therefore, EM may be an efficient tool that can be embedded in a large scale neural network algorithm.
- Because we have observed that shrinking the feasible region speeds up the proposed method, an interesting study might be to examine EM with some partitioning methods.
- For solving nonlinear constrained optimization problems, we have provided preliminary results on using penalty and barrier methods. Another well known

approach is Lagrangian methods. Further study along this line might produce promising results.

- Parallel algorithms are in their early stage of development. Recently, some good results with parallelization of stochastic methods have been reported [Ingber, 1994]. Because of its flexible structure, EM can be easily adapted to parallel algorithms. Moreover, having multiple populations at every iteration, with migration of points between them, might be explored.
- EM can be used as a preprocessing tool for providing feasible solutions to other methods that work within the confines of the feasible region such as interior point methods [Fang and Puthenpura, 1993, Nesterov and Nemirovski, 1994].
- There are certain problems for which no analytical form of the objective function exists. In this case, a usual approach to solve the problem is to sample points by Monte Carlo simulation. EM can be useful for solving these problems. The set of points from Monte Carlo simulation may constitute the population in EM at each iteration. Hence, the next set of points for optimizing the objective function can be generated by EM.

Bibliography

- E. H. L. Aarts and J. H. M. Korst. *Simulated Annealing and Boltzmann Machines*. Wiley, Chichester, 1997.
- E. H. L. Aarts and J. K. Lenstra, editors. *Local Search in Combinatorial Optimization*. Wiley, Chichester, 1997.
- M. M. Ali and C. Storey. Topographical Multi Level Single Linkage. *Journal of Global Optimization*, (5):349–358, 1994.
- R. S. Anderssen and P. Bloomfield. Properties of the random search in global optimization. *Journal of Optimization Theory and Applications*, (16):383–398, 1975.
- G. S. Anroulakis and M. N. Vrahatis. Optac: A portable software package for analyzing and comparing optimization methods by visualization. *Journal of Computational and Applied Mathematics*, (72):41–62, 1996.
- L. Armijo. Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific Journal of Mathematics*, (16):1–3, 1996.
- N. Baba. Convergence of a random optimization method for constrained optimization problems. *Journal of Optimization Theory and Applications*, 33(4):451–461, 1981.

- N. Baba, T. Shoman, and Y. Sawaragi. A modified convergence theorem for a random optimization method. *Information Sciences*, (13):159–166, 1977.
- R. Battiti and G. Tecchiolli. The continuous reactive tabu search: Blending combinatorial optimization and stochastic search for global optimization. *Annals of Operations Research*, (63):153–188, 1996.
- H. P. Benson. Concave minimization: Theory, applications and algorithms. In R. Horst and P. M. Pardalos, editors, *Handbook of Global Optimization*, volume 2 of *Nonconvex Optimization and Its Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.
- H. C. P. Berbee, C. G. E. Boender, A. H. G. Rinooy Kan, R. L. Smith C. L. Scheffer, and J. Telgen. Hit-and-run algorithms for the identification of nonredundant linear inequalities. *Mathematical Programming*, (37):184–207, 1987.
- D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Computer Science and Applied Mathematics. Academic Press, New York, NY, 1982.
- D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, Massachusetts, 1995.
- R. S. G. Beveridge. *Optimization: Theory and Practice*. McGraw-Hill, New York, 1970.
- Ş. İ. Birbil. The capacitated lot sizing problem (CLSP). Master’s thesis, Yeditepe University, Istanbul, Turkey, 1999.
- Ş. İ. Birbil and S. -C. Fang. An electromagnetism-like mechanism for global optimization. *to appear in Journal of Global Optimization*, 2000a.

- Ş. İ. Birbil and S. -C. Fang. A new heuristic for global optimization. In *Proceedings of the 8th Bellman Continuum*, pages 352–357, 2000b.
- Ş. İ. Birbil, S. -C. Fang, and R. L. Sheu. On the convergence of a population based global optimization algorithm. *submitted to Journal of Global Optimization*, 2001.
- Ş. İ. Birbil, L. Özdamar, M. Demirhan, and L. Helvacıoğlu. Computational experiments with probabilistic search methods in global optimization. Technical report, Yeditepe University, Istanbul, Turkey, 1999.
- C. G. E. Boender, A. H. G. Rinoooy Kan, and G. T. Timmer. A stochastic approach to global optimization. *Computational Mathematical Programming*, (F15):291–308, 1985.
- C. G. E. Boender and H. E. Romeijn. Stochastic methods. In R. Horst and P. M. Pardalos, editors, *Handbook of Global Optimization*, volume 2 of *Nonconvex Optimization and Its Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.
- C. A. Botsaris. A curvilinear optimization method based upon iterative estimation of the eigensystem of the hessian matrix. *Journal of Mathematical Analysis and Applications*, (63):396–411, 1978.
- R. P. Brent. *Algorithms for minimization without derivatives*. Prentice-Hall, Englewood Cliffs, NJ, 1973.
- S. H. Brooks. A discussion of random methods for seeking maxima. *Operations Research*, (6):244–251, 1958.
- B. G. O. Caprani and K. Madsen. Use of real-valued local minimum in parallel interval global optimization. *Interval Computations*, (3):71–82, 1993.

- E. W. Cowan. *Basic Electromagnetism*. Academic Press, New York, 1968.
- T. Csendes and J. Pintér. The impact of accelerating tools on the interval subdivision algorithm for global optimization. *European Journal of Operational Research*, (65): 314–320, 1993.
- L. Davis. *Genetic Algorithms and Simulated Annealing*. Pitman, London, 1987.
- A. Dekkers and E. H. L. Aarts. Global optimization and simulated annealing. *Mathematical Programming*, (50):367–393, 1991.
- M. Demirhan, L. Özdamar, L. Helvacioğlu, and Ş. İ. Birbil. FRACTOP: A geometric partitioning metaheuristic for global optimization. *Journal of Global Optimization*, (14): 415–435, 1999.
- L. Devroye. Progressive global random search of continuous functions. *Mathematical Programming*, (15):330–342, 1978.
- I. Diener. Trajectory nets connecting all critical points of a smooth function. *Mathematical Programming*, (36):340–352, 1986.
- I. Diener. Trajectory methods in global optimization. In R. Horst and P. M. Pardalos, editors, *Handbook of Global Optimization*, volume 2 of *Nonconvex Optimization and Its Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.
- L. C. W. Dixon and G. P. Szegö, editors. *Towards Global Optimization 1*. North-Holland, Amsterdam, 1975.
- L. C. W. Dixon and G. P. Szegö, editors. *Towards Global Optimization 2*. North-Holland, Amsterdam, 1978.

- V. M. M. Dorigo and A. Colomi. The Ant System: Optimization by a colony of cooperating agents. *IEEE Transactions on System, Man and Cybernetics*, (26):1–13, 1996.
- S. -C. Fang and S. Puthenpura. *Linear Optimization and Extensions*. Prentice Hall, Englewood Cliffs, NJ, 1993.
- R. Fletcher and C. Reeves. Function minimization by conjugate directions. *Computer Journal*, (7):149–154, 1964.
- C. A. Floudas. *Deterministic Global Optimization: Theory, Methods and Applications, 2nd Edition*, volume 37 of *Nonconvex Optimization and Applications*. Kluwer Academic Publishers, Dordrecht, Netherlands, 2 edition, 2000.
- C. A. Floudas and P. M. Pardalos. *A Collection of Test Problems for Constrained Global Optimization Algorithms*, volume 455. Springer-Verlag, Berlin, 1990.
- C. A. Floudas and P. M. Pardalos. *Recent Advances in Global Optimization*. Princeton University Press, 1992.
- C. A. Floudas, P. M. Pardalos, C. S. Adjiman, W. R. Esposito, Z. H. Gümüş, S. T. Harding, J. L. Klepeis, C. A. Meyer, and C. A. Schweiger. *Handbook of Test Problems in Local and Global Optimization*, volume 33 of *Nonconvex Optimization and Its Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1999.
- C. A. Floudas, P. M. Pardalos, C. S. Adjiman, W. R. Esposito, Z. H. Gümüş, S. T. Harding, J. L. Klepeis, C. A. Meyer, and C. A. Schweiger. Handbook of test problems in local and global optimization, 2002. <http://titan.princeton.edu/TestProblems/>.

- W. Forster. Homotopy methods. In R. Horst and P. M. Pardalos, editors, *Handbook of Global Optimization*, volume 2 of *Nonconvex Optimization and Its Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.
- L. R. Foulds. *Optimization Techniques: An Introduction*. Springer, New York, 1981.
- C. B. Garcia and W. I. Zangwill. *Pathways to Solutions, Fixed Points and Equilibria*. Prentice-Hall, Englewoods Cliffs, NJ, 1981.
- P. Gill and W. Murray. Quasi-newton methods for unconstrained optimization. *Institute of Mathematics and its Applications*, (9):91–108, 1972.
- F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, (13):533–549, 1986.
- F. Glover. Tabu search - Part I. *ORSA Journal on Computing* 1, 1(3):190–206, 1989.
- F. Glover. Tabu search - Part II. *ORSA Journal on Computing* 2, 2(1):4–32, 1990.
- F. Glover, J. P. Kelly, and M. Laguna. Genetic algorithms and tabu search: Hybrids for optimization. *Computers and Operations Research*, (22):111–134, 1995.
- F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1997.
- D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, New York, 1989.
- P. Hansen and B. Jaumard. Lipschitz optimization. In R. Horst and P. M. Pardalos, editors, *Handbook of Global Optimization*, volume 2 of *Nonconvex Optimization and Its Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.

- W. E. Hart. *Adaptive Global Optimization with Local Search*. PhD thesis, University of California, San Diego, 1994.
- R. Horst. A general class of branch and bound methods in global optimization with some new approaches for concave minimization. *Journal of Optimization Theory and Applications*, (51):271–291, 1986.
- R. Horst and P. M. Pardalos. *Handbook of Global Optimization*, volume 2 of *Nonconvex Optimization and Its Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.
- R. Horst, P. M. Pardalos, and N. V. Thoai. *Introduction to Global Optimization, 2nd Edition*, volume 48 of *Nonconvex Optimization and Its Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2 edition, 2000.
- R. Horst and H. Tuy. On the convergence of global methods in multiextremal optimization. *Journal of Optimization Theory and Applications*, (54):253–271, 1987.
- R. Horst and H. Tuy. *Global Optimization: Deterministic Approaches*. Springer Verlag, Berlin, 1993.
- W. Huyer and A. Neumaier. Global optimization by multilevel coordinate search. *Journal of Global Optimization*, (14):331–355, 1999.
- ICEO. International contest on evolutionary optimization (1st ICEO) web page, 2002. <http://iridia.ulb.ac.be/langerman/ICEO.html>.
- L. Ingber. Simulated annealing: Practice versus theory. *Journal of Mathematical Computation Modeling*, (18):29–57, 1994.

- D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, (79):157–181, 1993.
- R. B. Kearfott. An efficient degree-computation method for a generalized method of bisection. *Numerische Mathematics*, (32):109–127, 1979.
- R. B. Kearfott. *Rigorous Global Search: Continuous Problems*, volume 13 of *Nonconvex Optimization and Its Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.
- J. E. Kelley. The cutting plane method for solving convex programs. *SIAM Journal*, 8(4):703–712, 1960.
- A. Kirkpatrick, A. K. C. J. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, (220):671–680, 1983.
- B. Kosko. *Fuzzy Thinking (The New Science of Fuzzy Logic)*. Hyperion, New York, 1993.
- J. Kowalik and M. R. Osborne. *Methods for Unconstrained Optimization Problems*, volume 13 of *Modern Analytic and Computational Methods in Science and Mathematics*. Elsevier, Amsterdam, The Netherlands, 1968.
- P. J. M. Van Laarhoven and E. H. L. Aarts. *Simulated Annealing: Theory and Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1987.
- J. E. Lavery. Univariate cubic L_p splines and shape-preserving, multiscale interpolation by univariate cubic L_1 splines. *Computer Aided Geometric Design*, (17):319–336, 2000.

- A. M. Law and W. D. Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, 3rd Edition, New York, 2000.
- A. V. Levy, A. Montalvo, S. Gomez, and A. Calderon. *Topics in Global Optimization*, volume 909 of *Lecture Notes in Computer Science*. Springer, Berlin, 1981.
- D. G. Luenberger. *Introduction to Linear and Nonlinear Programming*. Addison Wesley, Massachusetts, 1973.
- MathWorks. MATLAB: The language of technical computing, 2000.
- Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Verlag, Berlin, 1994.
- K. G. Murty and S. N. Kabadi. Some NP-complete problems in quadratic and nonlinear programming. *Mathematical Programming*, (39):117–129, 1987.
- J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, (7):308–313, 1965.
- G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley and Sons, 1988.
- Y. E. Nesterov and A. Nemirovski. *Interior-Point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics, Philadelphia, 1994.
- A. Neumaier. *Interval Methods for Systems of Equations*. Cambridge University Press, Cambridge, 1990.
- A. Neumaier. Global optimization web page by Arnold Neumaier, 2002.
<http://solon.cma.univie.ac.at/neum/glopt.html>.

- I. H. Osman and J. P. Kelly. *Meta-Heuristics: Theory and Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.
- L. Özdamar and M. Demirhan. Experiments with new stochastic global optimization search techniques. (27):841–865, 2000. *Computers and Operations Research*.
- P. M. Pardalos and J. B. Rosen. Methods for global concave minimization: A bibliographic survey. *SIAM review*, (28):367–379, 1986.
- P. M. Pardalos and S. A. Vavasis. Quadratic programming with one eigenvalue is NP-hard. *Journal of Global Optimization*, (1):15–22, 1991.
- J. D. Pintér. Convergence properties of stochastic optimization procedures. *Math. Operationforsch. u. Statist.* , (15):405–427, 1984.
- J. D. Pintér. Convergence qualification of adaptive partitioning algorithms in global optimization. *Mathematical Programming*, (56):343–360, 1992.
- J. D. Pintér. Continuous global optimization software: A brief review. *Optima*, (52):1–8, 1996a.
- J. D. Pintér. *Global Optimization in Action, Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications*, volume 6 of *Nonconvex Optimization and Its Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996b.
- W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C*. Cambridge University Press, Cambridge, 1993.

- H. Ratschek and J. Rokne. Interval methods. In R. Horst and P. M. Pardalos, editors, *Handbook of Global Optimization*, volume 2 of *Nonconvex Optimization and Its Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.
- D. Ratz and T. Csendes. On the selection of subdivision directions in interval branch and bound methods for global optimization. *Journal of Global Optimization*, (7):183–207, 1995.
- A. H. G. Rinooy Kan and G. T. Timmer. Stochastic global optimization methods Part I: Clustering methods. *Mathematical Programming*, (39):27–56, 1987a.
- A. H. G. Rinooy Kan and G. T. Timmer. Stochastic global optimization methods Part II: Multi level methods. *Mathematical Programming*, (39):57–78, 1987b.
- A. H. G. Rinooy Kan and G. T. Timmer. Argument for unsolvability of global optimization problems. In *New Methods in Optimization and Their Industrial Uses*, pages 133–155. Birkhauser Verlag, Basel, 1989.
- A. H. G. Rinooy Kan and G. T. Timmer. Stochastic methods for global optimization. *American Journal of Mathematical Management Science*, (4):7–40, 1994.
- D. E. Rumelhart and J. L. McClelland, editors. *Parallel distributed processing*, volume 1. MIT Press, Cambridge, MA, 1986.
- J. D. Schaffer. A study of control parameters affecting online performance of genetic algorithms for function optimization. In *Proceedings of the 3rd International Conference on Genetic Algorithms*, pages 51–60, 1989.

- Y. Shang. *Global Search Methods for Solving Nonlinear Optimization Problems*. PhD thesis, University of Illinois at Urbana-Champaign, Urbana, Illinois, 1997.
- R. L. Smith. Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions. *Operations Research*, (32):1296–1308, 1984.
- F. J. Solis and R. J-B. Wets. Minimization by random search techniques. *Mathematics of Operations Research*, (6):19–30, 1981.
- M. Srinivas and L. Patnaik. Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Transactions on System, Man and Cybernetics*, (24):656–667, 1994.
- F. Stenger. Computing the topological degree of a mapping in \mathbb{R}^n . *Numerische Mathematik*, (25):23–38, 1975.
- A. E. Taylor and W. R. Mann. *Advanced Calculus*. John Wiley and Sons Inc., New York, 2 edition, 1972.
- H. M. Taylor and S. M. Karlin. *An Introduction to Stochastic Modeling*. Academic Press, San Diego, 1998.
- N. V. Thoai. A modified version of Tuy’s method for solving D.C. programming problems. *Optimization*, (19):665–674, 1988.
- A. Törn, M. M. Ali, and S. Viitanen. Stochastic global optimization: Problem classes and solution techniques. *Journal of Global Optimization*, (14):437–447, 1999.
- A. Törn and S. Viitanen. Topographical global optimization using pre-sampled points. *Journal of Global Optimization*, (5):267–276, 1994.

- A. Törn and A. Zilinskas. *Global Optimization*. Springer Verlag, Berlin, 1989.
- H. Tuy. Convex programs with an additional reverse convex constraint. *Journal of Optimization Theory and Applications*, (52):463–485, 1987a.
- H. Tuy. Global minimization of the difference of two convex functions. *Mathematical Programming Study*, (30):150–182, 1987b.
- L. Watson and W. H. Yang. Optimal design by a homotopy method. *Applicable Analysis*, (10):275–284, 1980.
- T. Westerlund and F. Pettersson. An extended cutting plane method for solving convex MINLP problems. *Computers and Chemical Engineering*, (19):131–136, 1995.
- M. Wodrich and G. Bilchev. Cooperative distributed search: The Ant's way. *Journal of Control and Cybernetics*, (26):3, 1997.
- G. R. Wood. Multidimensional bisection and global optimization. *Computer and Mathematics with Applications*, (21):161–172, 1991.
- S. J. Yakowitz and L. Fisher. On sequential search for the maximum of an unknown function. *Journal of Math. Anal. and Appl.*, (41):234–259, 1973.
- Z. B. Zabinsky and R. L. Smith. Pure adaptive search in global optimization. *Mathematical Programming*, (53):323–338, 1992.
- A. A. Zhigljavsky. *Theory of Global Random Search*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1991.
- H. J. Zimmermann. *Fuzzy Sets Theory and Its Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1990.

Appendix A

An Example Run of EM

We use the *Goldstein Price*¹ function to demonstrate a typical run of the method. This function achieves a global minimum of value 3 at (0, -1). The graph of the function is given in Figure A.1. In the following figures, \diamond represents the current best point and $*$ shows the location of the global optimum.

Figure A.2 shows the location of the points when the algorithm is started by randomly sampling points from the feasible region. Initially the current best point (x^{best}) is far away from the global optimum.

In Figure A.3, one of the points observes an objective function value better than the current best point, and this new point becomes the current best point. Thus, the points start to converge towards this new x^{best} .

The points in the population move towards the region around the current best point. Note that some points are repelled closer to the global optimum (Figure A.4). Figure A.5 demonstrates that the points in the population are directed towards the region around the current best point. And after this iteration, the global optimum is located.

¹We also include this function in our computational study, Section 3.2.2.

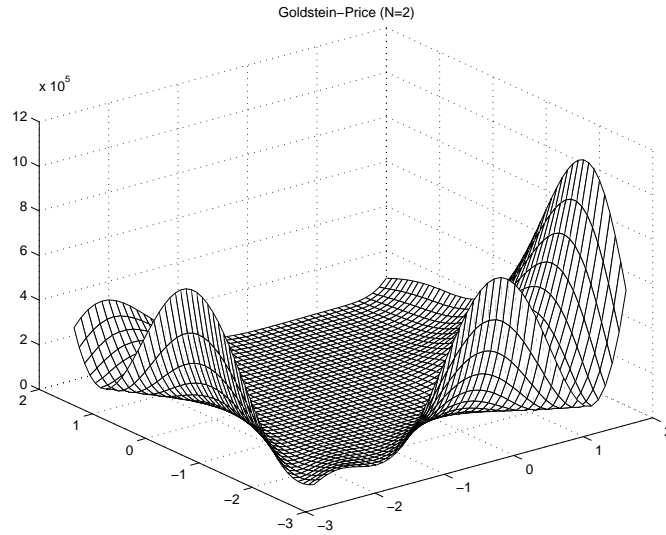


Figure A.1: Goldstein Price function

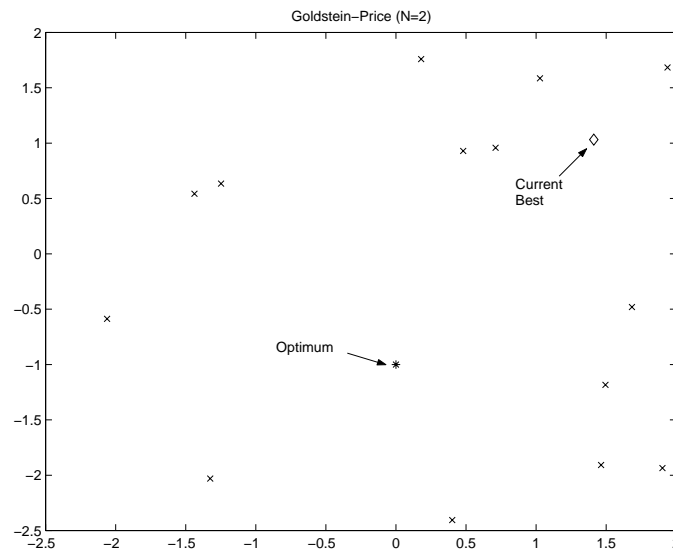


Figure A.2: The starting position of the particles just after the procedure *Initialize*.

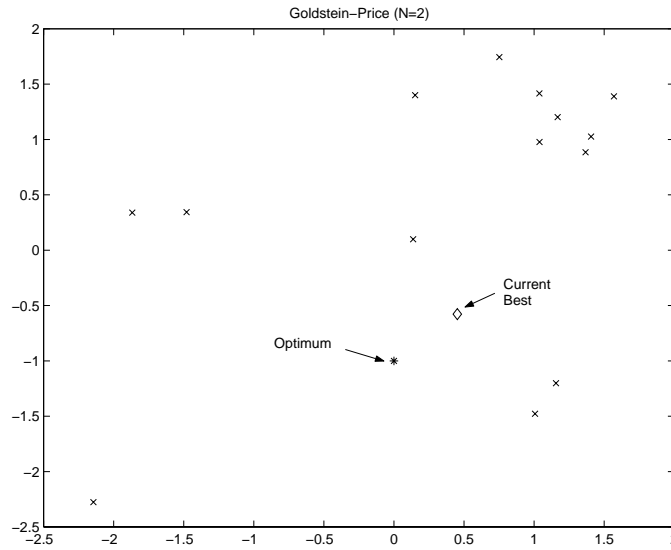


Figure A.3: A new point with better objective value.

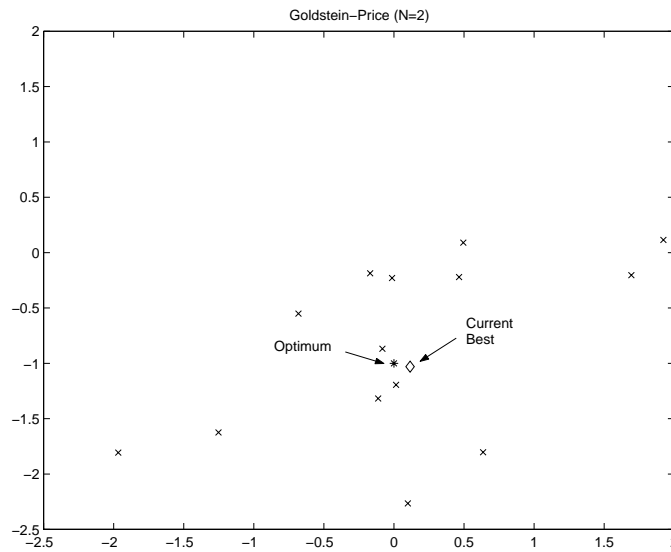


Figure A.4: Points start to attract and repel each other.

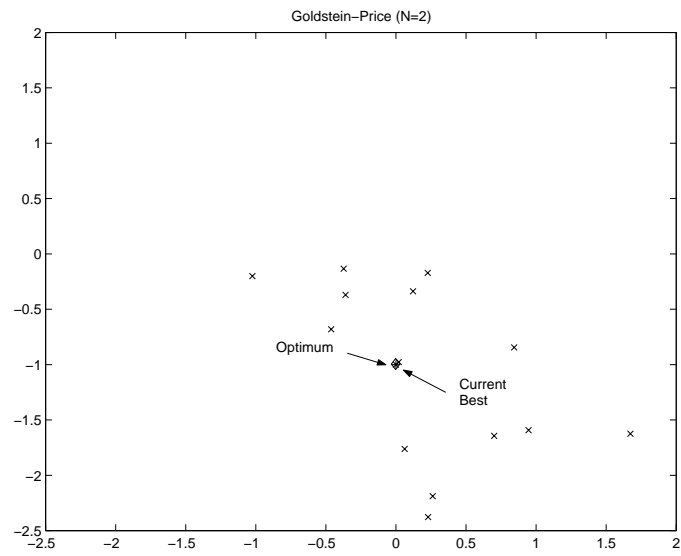


Figure A.5: Optimum is found. Points move toward the current best point

Appendix B

Test Functions

B.1 General Test Functions

Complex [Press et al., 1993]

$$f(x) = (x_1^3 - 3x_1x_2^2 - 1)^2 + (3x_1^2x_2 - x_2^3)^2$$

$$-2 \leq x_i \leq 2, \quad i = 1, 2$$

$$X^* = \left\{ (1, 0), \left(-\frac{1}{2}, \sqrt{\frac{1}{2}}\right), \left(-\frac{1}{2}, \sqrt{\frac{3}{2}}\right) \right\}$$

$$f(x^*) = 0$$

Davis [Davis, 1987]

$$f(x) = (x_1^2 + x_2^2)^{0.25} \left[\sin^2 (50(x_1^2 + x_2^2)^{0.1}) + 1.0 \right]$$

$$-100 \leq x_i \leq 100, \quad i = 1, 2$$

$$X^* = \{(0, 0)\}$$

$$f(x^*) = 0$$

Epistacity (4) [Srinivas and Patnaik, 1994]

$$f(x) = \sum_{i=1}^4 \frac{\sin(\pi k x_i)}{\pi k x_i}$$

$$-0.5 \leq x_i \leq 0.5, \quad i = 1, \dots, 4$$

$$X^* = \{(0, 0, 0, 0)\}$$

$$f(x^*) = 0$$

Epistacity (5) [Srinivas and Patnaik, 1994]

$$f(x) = \sum_{i=1}^5 \frac{\sin(\pi k x_i)}{\pi k x_i}$$

$$-0.5 \leq x_i \leq 0.5, \quad i = 1, \dots, 5$$

$$X^* = \{(0, 0, 0, 0, 0)\}$$

$$f(x^*) = 0$$

Griewank [ICEO, 2002]

$$f(x) = \frac{1}{200}(x_1^2 + x_2^2) - \cos x_1 \cos \frac{x_2}{\sqrt{2}} + 1$$

$$-100 \leq x_i \leq 100, \quad i = 1, 2$$

$$X^* = \{(0, 0)\}$$

$$f(x^*) = 0$$

Himmelblau [Botsaris, 1978]

$$f(x) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$$

$$-6 \leq x_i \leq 6, \quad i = 1, 2$$

$$X^* = \{(3, 2), (-2.805118, 3.131312), (3.584428, -1.848126), (-3.779310, -3.283186)\}$$

$$f(x^*) = 0$$

Kearfott [Kearfott, 1979]

$$f(x) = (x_1^2 - x_2^2)^2 + (x_2^2 - x_3^2)^2 + (x_3^2 - x_4^2)^2 + (x_4^2 - x_1^2)^2$$

$$-3 \leq x_i \leq 10, \quad i = 1, \dots, 4$$

$$X^* = \{(0, 0, 0, 0)\}$$

$$f(x^*) = 0$$

Levy [Levy et al., 1981]

$$f(x) = \sin^2\left(\pi\left(1 + \frac{x_1 - 1}{4}\right)\right) + \sum_{i=2}^9 \left(\frac{x_{i-1} - 1}{4}\right)^2 \left[1 + 10 \sin^2\left(\pi\left(1 + \frac{x_i - 1}{4}\right)\right)\right] + \left(\frac{x_{10} - 1}{4}\right)^2$$

$$-10 \leq x_i \leq 10, \quad i = 1, \dots, 10$$

$$X^* = \{(0, 0, 0, 0, 0, 0, 0, 0, 0, 0)\}$$

$$f(x^*) = 0$$

Rastrigin [MathWorks, 2000]

$$f(x) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2)$$

$$-5.12 \leq x_i \leq 5.12, \quad i = 1, 2$$

$$X^* = \{(0, 0)\}$$

$$f(x^*) = -2$$

Sine Envelope [Demirhan et al., 1999]

$$f(x) = 0.5 + \frac{\sin^2 \sqrt{x_1^2 + x_2^2} - 0.5}{(1.0 + 0.001(x_1^2 + x_2^2))^2}$$

$$-0.5 \leq x_i \leq 0.5, \quad i = 1, 2$$

$$X^* = \{(0, 0)\}$$

$$f(x^*) = 0$$

Stenger [Stenger, 1975]

$$f(x) = (x_1^2 - 4x_2)^2 + (x_2^2 - 2x_1 + 4x_2)^2$$

$$-1 \leq x_i \leq 4, \quad i = 1, 2$$

$$X^* = \{(0, 0), (1.695415, 0.7186082)\}$$

$$f(x^*) = 0$$

Step [Neumaier, 2002]

$$f(x) = \sum_{i=1}^5 [x_i]$$

$$-5.12 \leq x_i \leq 5.12, \quad i = 1, \dots, 5$$

$$X^* = \{([-5.12, -5]), [-5.12, -5]), [-5.12, -5]), [-5.12, -5]), [-5.12, -5])\}$$

$$f(x^*) = 0$$

Spiky [Michalewicz, 1994]

$$f(x) = -21.5 - x_1 \sin(4\pi x_1) - x_2 \sin(20\pi x_2)$$

$$-3 \leq x_1 \leq 12.1, \quad 4.1 \leq x_2 \leq 5.8$$

$$X^* = \{(11.62523, 5.72082)\}$$

$$f(x^*) = -38.85$$

Trid (5) [Neumaier, 2002]

$$f(x) = \sum_{i=1}^5 (x_i - 1)^2 - \sum_{i=2}^5 x_i x_{i-1}$$

$$-25 \leq x_i \leq 25, \quad i = 1, \dots, 5$$

$$X^* = \{(5, 8, 9, 8, 5)\}$$

$$f(x^*) = -30$$

Trid (20) [Neumaier, 2002]

$$f(x) = \sum_{i=1}^{20} (x_i - 1)^2 - \sum_{i=2}^{20} x_i x_{i-1}$$

$$-25 \leq x_i \leq 25, \quad i = 1, \dots, 20$$

$$X^* = \{(20, 38, 54, 68, 80, 90, 98, 104, 108, 110, 110, 108, 104, 98, 90, 80, 68, 54, 38, 20)\}$$

$$f(x^*) = -1520$$

B.2 Dixon and Szegö Test Functions**Shekel [S5], [S7], [S10] [Dixon and Szegö, 1975]**

$$f(x) = - \sum_{j=1}^K \frac{1}{\sum_{i=1}^4 (x_i - a_{ij})^2 + c_j}$$

$$K = \left\{ \begin{array}{ll} 5, & \text{for [S5]} \\ 7, & \text{for [S7]} \\ 10, & \text{for [S10]} \end{array} \right\}$$

j	a_{1j}	a_{2j}	a_{3j}	a_{4j}	c_j
1	4.0	4.0	4.0	4.0	0.1
2	1.0	1.0	1.0	1.0	0.2
3	8.0	8.0	8.0	8.0	0.2
4	6.0	6.0	6.0	6.0	0.4
5	3.0	7.0	3.0	7.0	0.4
6	2.0	9.0	2.0	9.0	0.6
7	5.0	5.0	3.0	3.0	0.3
8	8.0	1.0	8.0	1.0	0.7
9	6.0	2.0	6.0	2.0	0.5
10	7.0	3.6	7.0	3.6	0.5

$$0 \leq x_i \leq 10, \quad i = 1, \dots, 4$$

$$X^* = \{(4, 4, 4, 4)\} \text{ for [S5], [S7], and [S10]}$$

$$f(x^*) = \begin{cases} -10.1532, & \text{for [S5]} \\ -10.4029, & \text{for [S7]} \\ -10.5364, & \text{for [S10]} \end{cases}$$

Hartman [H3] [Dixon and Szegö, 1975]

$$f(x) = - \sum_{j=1}^4 (c_j e^{-\sum_{i=1}^3 a_{ij}(x_i - p_{ij})^2})$$

j	a_{1j}	a_{2j}	a_{3j}	p_{1j}	p_{2j}	p_{3j}	c_j
1	3.0	10.0	30.0	0.36890	0.11700	0.26730	1.0
2	0.1	10.0	35.0	0.46990	0.43870	0.74700	1.2
3	3.0	10.0	30.0	0.10910	0.87320	0.55470	3.0
4	0.1	10.0	35.0	0.03815	0.57430	0.88280	3.2

$$0 \leq x_i \leq 1, \quad i = 1, \dots, 3$$

$$X^* = \{(0.1, 0.55592, 0.85218)\}$$

$$f(x^*) = -3.8628$$

Hartman [H6] [Dixon and Szegö, 1975]

$$f(x) = - \sum_{j=1}^4 c_j e^{-\sum_{i=1}^6 a_{ij}(x_i - p_{ij})^2}$$

j	a_{1j}	a_{2j}	a_{3j}	a_{4j}	a_{5j}	a_{6j}	p_{1j}	p_{2j}	p_{3j}	p_{4j}	p_{5j}	p_{6j}	c_j
1	10.0	3.0	17.0	3.5	1.7	8.0	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886	1.0
2	0.05	10.0	17.0	0.1	8.0	14.0	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991	1.2
3	3.0	3.5	1.7	10.0	17.0	8.0	0.2348	0.1451	0.3522	0.2883	0.3047	0.6650	3.0
4	17.0	8.0	0.05	10.0	0.1	14.0	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381	3.2

$$0 \leq x_i \leq 1, \quad i = 1, \dots, 6$$

$$X^* = \{(0.20169, 0.15001, 0.47687, 0.2753, 0.31165, 0.65730)\}$$

$$f(x^*) = -3.3224$$

Goldstein Price [GP] [Dixon and Szegö, 1975]

$$f(x) = (1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)) \cdot \\ (30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)) \\ -2 \leq x_i \leq 2, \quad i = 1, 2$$

$$X^* = \{(0, -1)\}$$

$$f(x^*) = 3$$

Branin [BR] [Dixon and Szegö, 1975]

$$f(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos x_1 + 10 \\ -5 \leq x_1 \leq 10, \quad 0 \leq x_2 \leq 15$$

$$X^* = \{(-\pi, 12.275), (\pi, 2.275), (9.42478, 2.475)\}$$

$$f(x^*) = 0.3979$$

Six Hump Camel [C6] [Dixon and Szegö, 1975]

$$f(x) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + 4(x_2^2 - 1)x_2^2 \\ -5 \leq x_i \leq 5, \quad i = 1, 2$$

$$X^* = \{(0.08983, -0.7126)\}$$

$$f(x^*) = -1.0316$$

Shubert [SHU] [Dixon and Szegö, 1975]

$$f(x) = \prod_{i=1}^n \left(\sum_{j=1}^5 j \cos((j+1)x_i + j) \right)$$

$$-10 \leq x_i \leq 10, \quad i = 1, 2$$

$$X^* = \left\{ \begin{array}{lll} (-7.08351, 4.85806), & (5.48286, 4.85806), & (4.85806, -7.08351), \\ (4.85806, 5.48286), & (-7.08351, -7.70831), & (-7.70831, -7.08351), \\ (-1.42512, -0.80032), & (-0.80032, -1.42512), & (-1.42512, -7.08351), \\ (-7.08351, -1.42512), & (-7.70831, 5.48286), & (5.48286, -7.70831), \\ (5.48286, 4.85806), & (4.85806, 5.48286), & (-7.70831, -0.80032), \\ (-0.80032, -7.70831), & (-1.42512, -0.80032), & (-0.80032, -1.42512) \end{array} \right\}$$

$$f(x^*) = -186.7309$$

B.3 Hard Test Functions**Perm(4, 0.005) [Neumaier, 2002]**

$$f(x) = \sum_{k=1}^4 \left(\sum_{i=1}^4 (i^k + 0.005) \left(\left(\frac{x_i}{i} \right)^k - 1 \right) \right)^2$$

$$-4 \leq x_i \leq 4, \quad i = 1, \dots, 4$$

$$X^* = \{(1, 2, 3, 4)\}$$

$$f(x^*) = 0$$

Perm0(10, 100) [Neumaier, 2002]

$$f(x) = \sum_{k=1}^4 \left(\sum_{i=1}^{10} (i + 100) \left(x_i^k - \left(\frac{1}{i} \right)^k \right) \right)^2$$

$$-1 \leq x_i \leq 1, \quad i = 1, \dots, 10$$

$$X^* = \left\{ \left(1, 0.5, \frac{1}{3}, 0.25, 0.2, \frac{1}{6}, \frac{1}{7}, 0.125, \frac{1}{9}, 0.1 \right) \right\}$$

$$f(x^*) = 0$$

Powersum (8) [Neumaier, 2002]

$$f(x) = \sum_{k=1}^8 \left(\sum_{j=1}^8 x_j^k - \sum_{j=1}^8 \left(\frac{1}{j}\right)^k \right)^2$$

$$0 \leq x_i \leq 2, \quad i = 1, \dots, 8$$

$$X^* = \left\{ \left(1, 0.5, \frac{1}{3}, 0.25, 0.2, \frac{1}{6}, \frac{1}{7}, 0.125 \right) \right\}$$

$$f(x^*) = 0$$

Powersum (64) [Neumaier, 2002]

$$f(x) = \sum_{k=1}^{64} \left(\sum_{j=1}^{64} x_j^k - \sum_{j=1}^{64} \left(\frac{1}{j}\right)^k \right)^2$$

$$0 \leq x_i \leq 2, \quad i = 1, \dots, 64$$

$$X^* = \left\{ (x_1^*, \dots, x_i^*, \dots, x_{64}^*) \right\}, \quad x_i^* = \frac{1}{i}, \quad i = 1, \dots, 64$$

$$f(x^*) = 0$$

B.4 Nonsmooth Convex Test Functions**General Form of Cubic L_1 Functions [Lavery, 2000]**

$$f(x) = \sum_{i=1}^{n-1} \left\{ \begin{array}{ll} \frac{3}{2} |x_i + x_{i+1} - 2\Delta z_i| + \frac{(x_{i+1} - x_i)^2}{6|x_i + x_{i+1} - 2\Delta z_i|} + e^{-4} |x_i|, & \text{if } |x_{i+1} - x_i| \leq \\ 3 |x_i + x_{i+1} - 2\Delta z_i| - e^{-8}, & \\ |x_{i+1} - x_i| + e^{-4} |x_i|, & \text{otherwise} \end{array} \right\}$$

$$+ e^{-4} |x_n|$$

$$\Delta z_i = \frac{z_{i+1} - z_i}{y_{i+1} - y_i}, \quad i = 1, \dots, n$$

F1 (n=10)

i	1	2	3	4	5	6	7	8	9	10
z_i	0	0	0	0	0	1	0	0	0	0
y_i	0	1	2	3	4	4.1	5.1	6.1	7.1	8.1

$$X^* = \{(0, 0, 0, 0, 0, -0.000000335, 0, 0, 0, 0)\}$$

$$f(x^*) = 33$$

F2 (n=10)

i	1	2	3	4	5	6	7	8	9	10
z_i	0	0	0	0	1	1	0	0	0	0
y_i	0	1	2	3	3.1	4.1	5.1	6.1	7.1	8.1

$$X^* = \{(0, 0, 0, -0.000000004, 3.818181328, -1.354838549, \\ -0.000000166, 0.000000012, -0.000000002, 0.000000001)\}$$

$$f(x^*) = 31.3642$$

F3 (n=13)

i	1	2	3	4	5	6	7	8	9	10	11	12	13
z_i	0	0	0	1	1	1	1	1	1	1	0	0	0
y_i	0	1	2	3	4	5	6	7	8	9	10	11	12

$$X^* = \{(0, 0, 0, 0.000000664, -0.000000141, 0.000000196, \\ 0.000000207, 0.000000126, 0.000000028, 0, 0, 0, 0)\}$$

$$f(x^*) = 24.1801$$

F4 (n=13)

i	1	2	3	4	5	6	7	8	9	10	11	12	13
z_i	0	0	0	2	-4	6	-8	10	-12	5	5	5	5
y_i	0	1	2	3	4	5	5.1	7	8	9	10	11	12

$$X^* = \{(0, 0, 0, 0.000000664, -0.000000141, 0.000000196, \\ 0.000000207, 0.000000126, 0.000000028, 0, 0, 0, 0)\}$$

$$f(x^*) = 619.4210$$

B.5 Constrained Test Problems**TP 1 [Floudas et al., 1999]**

$$\min_x f(x) = 37.293239x_1 + 0.8356891x_1x_5 + 5.3578547x_3^2 - 40792.141$$

s.t.

$$-0.0022053x_3x_5 - 0.0056858x_2x_5 + 0.0006262x_1x_4 - 6.665593 \leq 0$$

$$0.0022053x_3x_5 - 0.0056858x_2x_5 - 0.0006262x_1x_4 - 83.334407 \leq 0$$

$$0.0071317x_2x_5 + 0.0021813x_3^2 + 0.0029955x_1x_2 - 29.48751 \leq 0$$

$$-0.0071317x_2x_5 - 0.0021813x_3^2 - 0.0029955x_1x_2 + 9.48751 \leq 0$$

$$0.0047026x_3x_5 + 0.0019085x_3x_4 + 0.0012547x_1x_3 - 15.699039 \leq 0$$

$$-0.0047026x_3x_5 - 0.0019085x_3x_4 - 0.0012547x_1x_3 + 10.699039 \leq 0$$

$$78 \leq x_1 \leq 102$$

$$33 \leq x_2 \leq 45$$

$$27 \leq x_3 \leq 45$$

$$27 \leq x_4 \leq 45$$

$$27 \leq x_5 \leq 45$$

$$27 \leq x_6 \leq 45$$

$$X^* = \{(78, 33, 29.9953, 45, 36.7758)\}$$

$$f(x^*) = -30665.5387$$

TP 2 [Floudas et al., 1999]

$$\min_x f(x) = -25(x_1 - 2)^2 - (x_2 - 2)^2 - (x_3 - 1)^2 - (x_4 - 4)^2 - (x_5 - 1)^2 - (x_6 - 4)^2$$

s.t.

$$(x_3 - 3)^2 + x_4 \geq 4$$

$$(x_5 - 3)^2 + x_6 \geq 4$$

$$x_1 - 3x_2 \leq 2$$

$$-x_1 + x_2 \leq 2$$

$$x_1 + x_2 \leq 6$$

$$x_1 + x_2 \geq 2$$

$$0 \leq x_1 \leq 6$$

$$0 \leq x_2 \leq 2$$

$$1 \leq x_3 \leq 5$$

$$0 \leq x_4 \leq 6$$

$$1 \leq x_5 \leq 5$$

$$0 \leq x_6 \leq 10$$

$$X^* = \{(5, 1, 5, 0, 5, 10)\}$$

$$f(x^*) = -310$$

TP 3 [Floudas et al., 1999]

$$\min_x f(x) = -x_1 - x_2$$

s.t.

$$2 + 2x_1^4 - 8x_1^3 + 8x_1^2 \geq x_2$$

$$4x_1^4 - 32x_1^3 + 88x_1^2 - 96x_1 + 36 \geq x_2$$

$$0 \leq x_1 \leq 3$$

$$0 \leq x_2 \leq 4$$

$$X^* = \{(2.3295, 3.17846)\}$$

$$f(x^*) = -5.50796$$

TP 4 [Floudas et al., 1999]

$$\min_x f(x) = 0.5x_1x_2^{-1} - x_1 - 5x_2^{-1}$$

s.t.

$$0.01x_2x_3^{-1} + 0.01x_1 + 0.0005x_1x_3 \leq 1$$

$$0 \leq x_1, x_2, x_3 \leq 100$$

$$X^* = \{(88.2890, 7.7737, 1.3120)\}$$

$$f(x^*) = -83.254$$

TP 5 [Floudas et al., 1999]

$$\min_x f(x) = -x_1 + 0.4x_1^{0.67}x_3^{-0.67}$$

s.t.

$$0.05882x_3x_4 + 0.1x_1 \leq 1$$

$$4x_2x_4^{-1} + 2x_2^{-0.71}x_4^{-1} + 0.05882x_2^{-1.3}x_3 \leq 1$$

$$0.1 \leq x_1, x_2, x_3, x_4 \leq 10$$

$$X^* = \{(8.1267, 0.6154, 0.5650, 5.6368)\}$$

$$f(x^*) = -5.7398$$