INTERVAL METHOD FOR SPECIAL CONSTRAINED
GLOBAL OPTIMIZATION PROBLEMS

by

MENGYI YING
MIN SUN, COMMITTEE CHAIR
WEI SHEN HSIA
JOSEPH NEGGERS
TAVAN TRENT
YANG XIAO

A DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the Department of Mathematics
in the Graduate School of
The University of Alabama

TUSCALOOSA, ALABAMA

2014
ABSTRACT

The basic framework in this dissertation is the standard Hansen method. The standard Hansen method has become a very well-known and popular method over the last few decades in the area of Interval Arithmetic [1] for finding all solutions of optimization problem.

Listed below are some strategies we have developed:

a). Binary tree data structure is proposed to record all the data associated with a functional relationship. We build a tree with the nodes representing all the operations and variables based upon the given function. Then we use a backward propagation method to obtain a new subdivision direction to be bisected. This method could accelerate the convergence after choosing the dominant variable in the tree data structure. Numerical results are illustrated to show the effectiveness in Chapter 3.

b). Affine arithmetic is proposed to reduce the approximations’ error and can be applied to linearize the given function and also applied in dealing with linear constraints. The main use of AA is to keep track of rounding errors for each computed value. Besides, AA provides a much tighter bound. Numerical results are illustrated to show the effectiveness in Chapter 3.

c). Exclusion zone functions are introduced to further reduce variables’ intervals which can be added in the deletion step in the standard Hansen algorithm. Numerical results are illustrated to show the effectiveness in Chapter 3.
d). For constrained global optimization problems with one linear constraint, we have developed a backward interval value detection method as noted in Chapter 4.

e). The four activities outlined above were performed to support this final activity. Three different procedures are developed (coordinate descent method, cutting line method, and projection coordinate descent method) to locate a feasible sampling point for the two linear constraints. Numerical results are illustrated to show the effectiveness in Chapter 5.

The three procedures referenced above are then combined with the Interval Arithmetic method [1] to compare against the standard Hansen method without using any proposed strategies. The results of all three procedures in combination with the Interval Arithmetic method [1] are found to be superior to the standard Hansen technique.
DEDICATION

This dissertation is dedicated to everyone who helped me and guided me through the trials of creating this manuscript. In particular, thanks to my advisor Dr. Sun Min, committee members and all friends who helped me throughout the time taken to complete this dissertation.
LIST OF ABBREVIATIONS AND SYMBOLS

GOP             Global Optimization Problem
C-GOP           Constrained Global Optimization Problem
$x_i$          The i-th Variable
V               Set of variables
X               Interval
$f(x)$         Objective Function
F(X)           Inclusion Function
$h_i$          The i-th Equation
$g_i$          The i-th Inequation
$H_i$          The i-th Equation’s Inclusion Function
$G_i$          The i-th Inequation’s Inclusion Function
$D_i$          $x_i$’s Domain
$D_i, lbX_i$    $x_i$’s Lower Bound
$\overline{D}_i, ubX_i$ $x_i$’s Upper Bound
$x^*$          The Optimal Minimizer
$f(x^*)$       The Optimal Objective Function Value
$\mathbb{R}$    The Set of Real Numbers
$\mathbb{I}$   The Set of Real Compact Intervals
$\mathbb{II}^n$ The Set of n-Dimensional Interval Vectors
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\omega(X)$</td>
<td>The Width of an Interval $X$</td>
</tr>
<tr>
<td>$\text{mid } X$</td>
<td>The Midpoint of an Interval $X$</td>
</tr>
<tr>
<td>$y$</td>
<td>The Lower Bound of The Inclusion Function</td>
</tr>
<tr>
<td>$\bar{x}$</td>
<td>The Feasible Point</td>
</tr>
<tr>
<td>$\bar{f}, f_{\text{best}}$</td>
<td>The Optimal Objective Function Value Used in Code</td>
</tr>
<tr>
<td>$=$</td>
<td>Equal</td>
</tr>
<tr>
<td>$&lt;$</td>
<td>Less Than</td>
</tr>
<tr>
<td>$V_{\text{max}}$</td>
<td>The Maximum Variation</td>
</tr>
<tr>
<td>$V_{\text{sum}}$</td>
<td>The Summation of Variations</td>
</tr>
</tbody>
</table>
ACKNOWLEDGMENTS

First and foremost, a sincere thanks is hereby extended to the following who never stopped in helping until this dissertation is constructed.

The committee chair and dissertation advisor, Dr. Sun Min, for the patient guidance;

Dr. Neggers Joseph, my friend Rob and Reena, for pointing out grammar mistakes and informal expressions;

Other members of the defense committees, Dr. Hsia Wei-Shen, Dr. Trent Tavan, Dr. Xiao Yang, for sharing their precious time and positive insights;

The staff in the Department of Mathematics, for offering an ideal environment for my teaching and research.

Last but not least, a special thanks to my friends and family, for all the support and encouragement.
## CONTENTS

ABSTRACT .................................................................................................................................... ii

DEDICATION ............................................................................................................................... iv

LIST OF ABBREVIATIONS AND SYMBOLS ........................................................................... v

ACKNOWLEDGMENTS ............................................................................................................ vii

LIST OF TABLES ....................................................................................................................... x

LIST OF FIGURES ................................................................................................................... xii

CHAPTER 1: INTRODUCTION ................................................................................................... 1

CHAPTER 2: BACKGROUND IN GLOBAL OPTIMIZATION AND INTERVAL METHOD. 5

2.1 Problem Statements: (C-GOP) and (GOP) ................................................................. 5

2.2 Interval Arithmetic and Inclusion Functions ............................................................. 7

2.3 A Standard Interval Method for (GOP): The Hansen Algorithm ......................... 9

2.4 Standard Constraint Treatment Methods .................................................................. 18

CHAPTER 3: IMPROVEMENT STRATEGIES FOR INTERVAL METHODS IN GLOBAL OPTIMIZATION .......................................................................................... 23

3.1 Binary Tree Data Structure ..................................................................................... 23

3.2 Affine Arithmetic ..................................................................................................... 30

3.3 Exclusion Zone Functions ...................................................................................... 35

CHAPTER 4: IMPROVED TREATMENT OF ONE SPECIAL CONSTRAINT ...................... 41

4.1 Introduction ............................................................................................................. 41
4.2 Algorithm with One Special Constraint ................................................................. 42
4.3 A Class of Examples with One Special Constraint .................................................. 47
4.4 Numerical Results: One Special Sonstraint Case .................................................... 48

CHAPTER 5: PROCEDURES DEALING WITH TWO LINEAR CONSTRAINTS .............. 52

5.1 Coordinate Decent method ..................................................................................... 52
5.2 Cutting Line method ............................................................................................... 61
5.3 Projection Coordinate Descent method ................................................................... 69
5.4 Data Comparison ...................................................................................................... 73

REFERENCES .............................................................................................................. 99

APPENDIX: LIST OF EXAMPLES ............................................................................... 104
LIST OF TABLES

Table 2.1 Detailed $f_{\text{best}}$ data results comparison in 4 different selection options..................15
Table 2.2 Detailed CPU Time data results comparison in 4 different selection options..............16
Table 2.3 Detailed iteration data results comparison in 4 different selection options..............17
Table 3.1 Comparison for example 201.........................................................................................26
Table 3.2 Detailed data results between Hansen and H-Dir..........................................................29
Table 3.3 Detailed data results between the Hansen and the AA method.................................34
Table 3.4 Zone functions and their coded IDs...............................................................................36
Table 3.5 Detailed data results between the Hansen and the Exclusion Zone method..............39
Table 4.1 Data comparison between the original algorithm and the modified algorithm.................................................................48
Table 5.1 Detailed $f_{\text{best}}$ data for four methods and the unconstrained case in 4 dimensions....76
Table 5.2 Detailed CPU Time data for four methods and the unconstrained case in 4 dimensions........................................................................................................................................77
Table 5.3 Detailed iteration data for four methods and the unconstrained case in 4 dimensions..78
Table 5.4 Feasibility for four methods in 4 dimensions.................................................................79
Table 5.5 Detailed $f_{\text{best}}$ data for four methods and the unconstrained case in 10 dimensions......82
Table 5.6 Detailed CPU Time data for four methods and the unconstrained case in 10 dimensions........................................................................................................................................83
Table 5.7 Detailed iteration data for four methods and the unconstrained case in 10 dimensions........................................................................................................................................84
Table 5.8 Feasibility for four methods in 10 dimensions..............................................................85
Table 5.9 Detailed $f_{\text{best}}$ data for four methods and the unconstrained case in 40 dimensions.......88
Table 5.10 Detailed CPU Time data for four methods and the unconstrained case in 40 dimensions……………………………………………………………………………………………………89

Table 5.11 Detailed iteration data for four methods and the unconstrained case in 40 dimensions………………………………………………………………………………………………90

Table 5.12 Feasibility for four methods in 40 dimensions……………………………………………………………………………………………………………………………91

Table 5.13 Detailed f_best data for four methods and the unconstrained case in 100 dimensions………………………………………………………………………………94

Table 5.14 Detailed CPU Time data for four methods and the unconstrained case in 100 dimensions………………………………………………………………………………………………95

Table 5.15 Detailed iteration data for four methods and the unconstrained case in 100 dimensions…………………………………………………………………………………………………96

Table 5.16 Feasibility for four methods in 100 dimensions…………………………………………………………………………………………………………………………………………97
LIST OF FIGURES

Figure 2.1 $f_{\text{best}}$ comparison for 4 different selection options.................................13
Figure 2.2 CPU time comparison for 4 different selection options.................................13
Figure 2.3 Iteration comparison for 4 different selection options.................................14
Figure 3.1 Binary Tree Data Structure for Example 201.................................................25
Figure 3.2 Tracing the lower bound in the Binary Tree Data Structure for Example 201.....26
Figure 3.3 $f_{\text{best}}$’s difference between Hansen and H-Dir..............................................27
Figure 3.4 CPU Time difference between Hansen and H-Dir.............................................28
Figure 3.5 Iteration difference between Hansen and H-Dir...............................................28
Figure 3.6 $f_{\text{best}}$’s difference between the Hansen and the AA method...........................32
Figure 3.7 CPU time difference between the Hansen and the AA method........................32
Figure 3.8 Iteration difference between the Hansen and the AA method...........................33
Figure 3.9 $f_{\text{best}}$’s difference between the Hansen and the Exclusion Zone method..........37
Figure 3.10 CPU time difference between the Hansen and the Exclusion Zone method......38
Figure 3.11 Iteration difference between the Hansen and the Exclusion Zone method......38
Figure 5.1 Coordinate descent in no conflict cases.........................................................54
Figure 5.2 Coordinate descent in conflict cases of type 1...............................................55
Figure 5.3 Coordinate descent in conflict cases of type 2a...............................................56
Figure 5.4 Coordinate descent in conflict cases of type 2b...............................................57
Figure 5.5 Coordinate descent in conflict cases of type 2c...............................................58
Figure 5.6 Coordinate descent in conflict cases of type 2d...............................................59
Figure 5.7 Coordinate descent in conflict cases of type 2e...........................................................60
Figure 5.8 Boundary region........................................................................................................65
Figure 5.9 Feasible line going through the boundary region......................................................71
Figure 5.10 $f_{\text{best}}$ comparison in 4 dimensions.................................................................74
Figure 5.11 Iteration comparison in 4 dimensions.....................................................................75
Figure 5.12 CPU Time comparison in 4 dimensions.................................................................75
Figure 5.13 $f_{\text{best}}$ comparison in 10 dimensions.................................................................80
Figure 5.14 Iteration comparison in 10 dimensions..................................................................80
Figure 5.15 CPU Time comparison in 10 dimensions.................................................................81
Figure 5.16 $f_{\text{best}}$ comparison in 40 dimensions.................................................................86
Figure 5.17 Iteration comparison in 40 dimensions..................................................................86
Figure 5.18 CPU Time comparison in 40 dimensions.................................................................87
Figure 5.19 $f_{\text{best}}$ comparison in 100 dimensions.................................................................92
Figure 5.20 Iteration comparison in 100 dimensions..................................................................92
Figure 5.21 CPU Time comparison in 100 dimensions..............................................................93
CHAPTER 1
INTRODUCTION

Optimization techniques are beneficial and widely used throughout the world to automatically examine different solutions and select the best desired choice. In mathematics, these techniques are called global optimization. The desired choice is called the global optimization solution.

The global optimization solution in this dissertation is to discover the global minimum. In general, the approach to reach the global minimum and all corresponding variables’ values is difficult and complicated. However, the Interval Arithmetic (IA) method has successfully demonstrated its effectiveness in the last century. It is a technique for numerical computation where each variable’s value and each iteration cycle are represented as an interval of floating numbers. Two major advantages of the IA method is that it insures all possible solutions and it produces a more accurate estimated bound of the solution to a given function. Recently, IA has been extended and used broadly and effectively. It is usually used in unconstrained global optimization problems (GOP).

Interval Arithmetic was firstly developed by Ramon E. Moore in the 1960’s. He wrote a book called “Interval Analysis” [7] and brought interval mathematics into the main stream. Moore concluded the previous error bounding work researched by others (see [8]) and extended into a useful tool (introduced in [15, 16]) for error analysis. In 1992, the team of Hansen and Walster wrote a book called “Global Optimization Using the Interval Analysis” [1] (this book
was revised in 2002). This book discussed some important methods without including constraints, such as the Moore-Skelboe Algorithm, and the Ichida-Fujii Algorithm. This book also discussed several methods involving constraints, such as the Newton method, penalty method. They expanded upon this to develop their own system called the Hansen algorithm and develop sharp bounds on interval polynomial roots [12].

A constrained global optimization problem (C-GOP) is more difficult to solve than a global optimization problem without restrictions (GOP). The most common and simplest approach to deal with C-GOP is to utilize the framework of the Interval Arithmetic (IA) method.

The IA method makes use of constrained inclusion functions to detect subdomains that do not contain any feasible solutions. There is a simple and useful technique called “interval branch and bound”. This technique is one of the successful methods for solving the GOP and C-GOP. However, there are some other search methods such as stochastic search methods have been more popular choices since they are easier to implement and faster to reach an approximate solution [52]. But stochastic methods cannot ensure convergence to a global solution when a particular run ends generally [52]. Even if a good approximate solution has been confronted when the search is processing, stochastic method do not have reliable technique to detect it [52]. This causes additional and unnecessary efforts including calculation and running time [52]. Therefore, these methods are usually used in GOP without constraints [52]. If we apply these methods in problems involving constraints, some justifications need to be added. Under the framework of interval brand and bound, a number of advantages can be listed as follows:

1. Convergence for all global solutions is guaranteed under weak assumptions, even in the presence of round-off errors [52].

2. Reliable stopping criteria is offered for the algorithm to stop when necessary [52].
3. It is numerically robust and round-off errors are handled conveniently and effectively [52].

4. It can be justified theoretically [52].

5. It handles constraints easily without jeopardizing theoretic justifications [52].

Because of the above advantages, a huge effort of research has been focused on developing IA methods including GOP and C-GOP. For GOP, there are Sunaga in [9], Domes and Neumaier in [10], Caprani, Godthaab, and Madsen in [28], Eriksson in [30, 31], Jansson and Knuppel in [32, 33, 34], Mayer. G in [43], Moore, Hansen, and Leclerc In [44], and A.Rum in [45]. For C-GOP, there are Hansen in [1], Domes and Neumaier in [2], Kearfott, Novoa, and M. III in [36] and [37], Kristinsdottir, Zabinskym, Csendes, Tuttle, and M.E. in [48], Wolfe in [46], Frederic Messine in [49], Yahia Lebbah, Claude Michel, and Michel Rueher in [51], H. Schichl in [11]. (see [27] for more details)

However, most of the examples demonstrated in the above papers seem to be applied in limited dimensional problems (much less than 100 dimensions) [52]. There are two major concerns in solving large dimensional problems: large amount of memory space and slow speed of convergence [52]. The number of sub-boxes to be processed in an interval method could potentially increase exponentially when the dimension of the domain increases [52]. Inspired by such observations, we have developed some new strategies associated with the interval branch and bound methodology [52]. Our new versions of the interval algorithm in this dissertation show improvement both in reducing memory space usage and speeding convergence [52].

This dissertation is organized as follows: In Chapter 2, we review the background of C-GOP and the basic interval arithmetic method. In Chapter 3, we apply some useful improvement strategies for our algorithm, such as Binary Tree Data Structure discussed in [21, 22], Affine
Arithmetic discussed in [3, 6, 19], and Exclusion Zone functions discussed in [17, 18, 23]. All numerical results are included to demonstrate their effectiveness. In addition, we develop several useful new techniques to locate the local feasible sampling points such as the idea discussed in [52]. In Chapter 4, two different versions of backward iterative scheme dealing with one special constraint are developed. Algorithms are presented along with theoretical convergence results. In Chapter 5, three different methods dealing with two linear constraints are introduced and compared. Numerical testing results are given in Section 5.4.
CHAPTER 2

BACKGROUND IN GLOBAL OPTIMIZATION AND INTERVAL METHOD

Many scientists, mathematicians, engineers, and investors want to find optimal solutions to their particular problems. Examples include determining the best method to maximize profit, reduce waste, and make the best choice within an environment filled with complicated variables. Normally, solutions must satisfy constraints that cannot be ignored. Therefore, optimization techniques have become important and widely used in industry and science. Here are the definitions for GOP and C-GOP in mathematics.

2.1 Problem Statements: (C-GOP) and (GOP)

**Definition 2.1:** A general global optimization problem (GOP) is defined to find the values of $n$ variables $x_1, x_2, x_3, ..., x_n$, which minimize the objective function $f(x)$.

$$\text{min } f(x_1, x_2, x_3, ..., x_n) \quad (2.1.1)$$

with variables’ domains $D_i = [D_{il}, D_{iu}]$ for $x_i$, $i = 1, ..., n$. We also call this statement an unconstrained global optimization problem.

**Definition 2.2:** A general constrained global optimization problem (C-GOP) is to minimize the objective function $f(x_1, x_2, x_3, ..., x_n)$ subject to the given constraints (equalities and inequalities). Thus it is defined by

a) A set of variables, $V = \{x_1, ..., x_n\}$;
b) A set of constraints, \(C = \{h_1, \ldots, h_p, g_1, \ldots, g_q\}\), over the variables \(x_i\) of the problem.

Note: Constraints can be linear or nonlinear equations or inequalities.

We could represent the statement of (C-GOP) as follows:

\[
\min f (x_1, x_2, x_3 \ldots x_n)
\]

subject to
\[
h_i(x_1, x_2, x_3 \ldots x_n) = 0, \quad i = 1, 2, \ldots, p,
\]
\[
g_j(x_1, x_2, x_3 \ldots x_n) \leq 0, \quad j = 1, 2, \ldots, q.
\]

with variables’ domains
\[
D_i = [D_i, \overline{D}_i] \text{ for } x_i, \quad i = 1, \ldots, n.
\]

Note: any set of values of the variables \(x = (x_1, \ldots, x_m)^T\) can be interpreted as a point in the m-dimensional space.

**Definition 2.3** [1]: A point \(x \in D\) satisfying the constraints, that is, (using the vector notation) \(g(x) \leq 0\) and \(h(x) = 0\) is called feasible point. A set \(U \in D\) is called a feasible set if all points of \(U\) are feasible. If the problem is unconstrained then there are no constraints and every point is feasible.

**Definition 2.4** [1]: A local minimum point or a local minimizer of C-GOP is feasible point \((x^*)\) of the search space \((X = D_1 \times D_2 \times \ldots \times D_n)\), that meets all the constraints, i.e. that satisfies all equations/inequalities simultaneously, and whose objective function value
\[
f(x^*) \leq f(x)
\]
for all feasible points \(x \in D\). The unconstrained version (GOP) is formulated similarly, i.e. without imposing any of the equations/inequalities constraints. (GOP) is also referred to as the box-constrained global optimization problem.

**Definition 2.5** [1]: The value of \(f(x^*)\) is called a local minimum or a local minimum value if

\[
f(x^*) \leq f(x) \quad \text{for all feasible points } x \in D.
\]
2.2 Interval Arithmetic and Inclusion Functions

In interval analysis, an unknown real variable $x$ is estimated by an interval $X$. This interval $X$ has an upper bound $ubX$ and a lower bound $lbX$ which encloses all possible values of $x$. Thus $x \in X$.

**Definition 2.6:** The set of real numbers is denoted by $\mathbb{R}$. Let $\mathbb{I}$ be the set of real compact intervals $[a, b]$. $a, b \in \mathbb{R}$. The set of n-dimensional interval vectors, also called boxes, is denoted by $\mathbb{I}^n$. Operations in $\mathbb{I}$ are defined by the expression

$$X \ast Y = \{ x \ast y : x \in X, y \in Y \} \text{ for } X, Y \in \mathbb{I}.$$  \hspace{1cm} (2.2.1)

where the symbol $\ast$ stands for arithmetic operations $\ast, -, \cdot, /$ ($X/Y$ is only defined when $0 \not\in Y$). $I(X)$ denotes the set of subintervals of $X$.

**Definition 2.6:** Simple constructive rules follow easily from the definition:

$$[a, b] + [c, d] = [a + c, b + d],$$

$$[a, b] - [c, d] = [a - d, b - c],$$

$$[a, b] \cdot [c, d] = [\min (ac, ad, bc, bd), \max (ac, ad, bc, bd)],$$  \hspace{1cm} (2.2.2)

$$[a, b] / [c, d] = [a, b] \cdot [1/d, 1/c], \text{ if } 0 \not\in [c, d].$$

**Example 2.1:**

1. $[1, 2] + [3, 4] = [4, 6];$
2. $[0,10] - [0,10] = [-10,10];$
3. $[-1,3] \cdot [-1/2, 1/2] = [-3/2, 3/2];$
4. $[1,2]/[1,2] = [1/2,2].$

**Property 2.1:** Subdistributive law:
\[ X \cdot (Y + Z) \subseteq X \cdot Y + X \cdot Z, \text{ for } X, Y, Z \in \mathbb{I}. \]  

(2.2.3)

**Property 2.2:** Inclusion isotonicity:

\[ X \subseteq Y, Z \subseteq T \rightarrow X \ast Z \subseteq Y \ast T, \text{ for } X, Y, Z, T \in \mathbb{I}. \]

(2.2.4)

**Definition 2.7:** The *width* of an interval \( X \) is denoted by

\[ \omega(X) = ub X - lb X. \]

(2.2.5)

and its *midpoint* is

\[ \text{mid } X = (ub X + lb X) / 2. \]

(2.2.6)

**Example 2.2:** Given the interval is \( X = [-1,2] \),

\[ \omega(X) = 2 - (-1) = 3, \]

\[ \text{mid } X = [2 + (-1)] / 2 = 1/2. \]

Given the interval vector \( X = (X_1, ..., X_n) \in \mathbb{I}^n \), its width is defined as

\[ \omega(X) = \max \{ \omega(X_i) : i = 1, ..., n \}. \]

(2.2.7)

And its *midpoint* is

\[ \text{mid } (X) = \left( \text{mid } (X_1), ..., \text{mid } (X_n) \right)^T. \]

(2.2.8)

**Definition 2.8:** The *Inclusion function* of \( f(x) \) denoted by \( F(X) \) is an interval-valued function from \( I(X) \) to \( I(\mathbb{R}) \) which encloses the range (also names the exact image) of a considered function \( f \) over any sub-box of \( X \). Therefore, one obtains:

\[ [\min_{x \in X} f(x), \max_{x \in X} f(x)] \subseteq F(X), \ \forall X \in I. \]

(2.2.9)

**Example 2.3:**

\[ f(x) = x - \frac{1}{2} x^2, \ x \in [0,1]. \]

The inclusion function is \( F(X) = [0,1] - \frac{1}{2} \times [0,1] \times [0,1] \).
\[
\begin{align*}
= [0,1] - \frac{1}{2} \times [0,1] \\
= [0,1] - [0.1/2] \\
= \left[-\frac{1}{2}, 1\right].
\end{align*}
\]

2.3 A Standard Interval Method for (GOP): The Hansen Algorithm

The standard Hansen method is considered as the first complete interval method for solving unconstrained global optimization problems [7]. Its convergence has been proved in [1]. Many enhanced versions have been developed since then. Enhancements were made in one or more steps of the standard prototype. One of them is using the constraints’ inclusion functions. This dissertation provides another enhancement to it. Before our new enhancement is described, a standard interval method for unconstrained global minimization is stated as follows [1]:

1. Input data: the dimension of the problem, \( n \); the domain \( X \) of the problem; inclusion function \( F \) of \( f \).
2. Set \( Y := X \), and let \( y := lb \ F(y) \).
3. If a feasible point \( \bar{x} \) is given, set \( \bar{f} := ub F(\bar{x}) \), else set \( \bar{f} := \infty \).
4. Initialize the list \( L = \{(Y, y)\} \).
5. Choose a box from the list \( L \), and remove it from the list.
6. Choose a coordinate direction for the splitting of the new \( Y \).
7. Bisect \( Y \) normal to the chosen direction, getting the boxes \( V_1, V_2 \), with corresponding lower bounds \( y_1, y_2 \).
8. For \( i = 1, 2 \)
a. Delete \((V_i, y_i)\) if it can be proven that \(V_i\) contains no optimal solution (e.g. when \(\bar{f} < y_i\));

b. Perform sampling at the midpoint and perform a possible update of \(\bar{f}\);

c. Store \((V_i, y_i)\) into the list if it is not deleted.

9. If one of the termination criteria holds, stop. Otherwise, go to Step 5.

For the above algorithm, we also added some features that will help the convergence.

2.3.1 The Standard Hansen method

The standard Hansen method uses the midpoint method in step 7. The midpoint method in mathematics is a root-locating method. It repeatedly bisects a domain and then selects a subdomain which a solution must lie inside. Although it is relatively time consuming, it is a very simple and robust method. Because of this property, it is often used to obtain a rough approximation to a solution. The method is derived based on the intermediate value theorem, and applicable when we solve the equation \(f(x) = 0\), where \(f\) is a continuous function on \(x \in [a, b]\) and \(f(a)\) and \(f(b)\) have opposite signs. In this dissertation, we shifted all our examples’ global minima, which could satisfy this condition to zero. In this case, the \(f\) must have at least one solution in the interval \([a, b]\).

At each step the method divides the interval in two subintervals \([a, c]\) and \([c, b]\) by computing the midpoint

\[
    c = (a + b) / 2, \quad (2.3.1)
\]

In interval notation it is
\[ c = \text{mid} \, X = (\text{ub} \, X + \, \text{lb} \, X) / 2. \] \hspace{1cm} (2.3.2)

For multiple variables, we have

\[ c = \text{mid} \, (X) = (\text{mid} \, (X_1), ..., \text{mid} \, (X_n))^T. \hspace{1cm} (2.3.3) \]

Thus we could obtain the function value \( f(c) \) at point \( c \). We also use this \( f(c) \) to update the objective function value \( f_{\text{best}} \) if we find \( f(c) < f_{\text{best}} \). The subintervals will be saved as new intervals in the list \( L \) waiting to be processed. In this approach all the subintervals will be guaranteed to be processed and all the solutions will ultimately be found when the interval reaches one of the stopping conditions, which will be discussed in the next section. We applied this feature in step 7 of the Hansen algorithm.

### 2.3.2 Stopping Conditions

In this dissertation, we develop several stopping conditions. They are as follows:

1. Maximum number of iterations. The number depends on the problems’ dimension. If it is a lower dimension such as a two variables’ problem, the maximum number of iterations can be set at 3000. The higher the dimension is, the bigger the maximum number of iterations should be.

2. Box size. The box refers to the variable’s domain. If the box size is small enough, it is not necessary to bisect the box.

3. Inclusion function’s range. If the inclusion function’s range is small enough, the optimal function value is found.

4. Maximum function call. In the algorithm, the number of the objective functions called is recorded. The more function calls, the less effective the algorithm is.
5. CPU time. CPU time is a measure to compare different methods. The less CPU time, the better the method is.

6. The optimal function value. This is called $f_{best}$. When the results generated from different methods are close enough to the known minimum objective function value, stop.

7. Empty list. The subinterval is added to the list $L$ if it doesn’t satisfy the deletion condition. But it is possible that all subintervals satisfy the deletion condition since the deletion condition is updated during the iterations. When the list $L$ is empty, stop.

2.3.3 Selection

In this dissertation, for step 5 and step 6 in the Hansen algorithm, we developed several options to select the box and direction.

For step 5, all the subintervals in the list $L$ need to be covered. In order to achieve this goal, we employ the four following methods:

1. Choose the lowest lower bound $y$ (called Lower-y). After comparing all the lower bound $y_i$, for $i = 1,2, ..., n$, we choose the lowest lower bound $y$.

2. Choose the first pair in the list $L$ (called Age).

3. Choose the smallest box size in the list $L$ (called BoxSize).

4. Choose the smallest size between the upper bound and lower bound (called $y$-Size).

It is difficult to tell which selection is better. Normally it depends on the problem we are dealing with. We present Figure 2.1-2.3 and Table 2.1-2.3 to demonstrate the effectiveness of selection mode.
Figure 2.1 $f_{\text{best}}$ comparison for 4 different selection options

Figure 2.2 CPU time comparison for 4 different selection options
Figure 2.3 Iteration comparison for 4 different selection options
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Lower-y</th>
<th>Age</th>
<th>BoxSize</th>
<th>y-Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>101</td>
<td>6.67E-05</td>
<td>6.67E-05</td>
<td>6.67E-05</td>
<td>6.67E-05</td>
</tr>
<tr>
<td>2</td>
<td>102</td>
<td>1.10E-06</td>
<td>1.10E-06</td>
<td>1.10E-06</td>
<td>1.10E-06</td>
</tr>
<tr>
<td>3</td>
<td>103</td>
<td>1.32E-06</td>
<td>1.32E-06</td>
<td>1.32E-06</td>
<td>1.32E-06</td>
</tr>
<tr>
<td>4</td>
<td>104</td>
<td>2.87E-05</td>
<td>2.87E-05</td>
<td>2.87E-05</td>
<td>2.87E-05</td>
</tr>
<tr>
<td>5</td>
<td>105</td>
<td>5.38E-05</td>
<td>5.38E-05</td>
<td>5.38E-05</td>
<td>5.38E-05</td>
</tr>
<tr>
<td>6</td>
<td>106</td>
<td>3.48E-05</td>
<td>3.48E-05</td>
<td>3.48E-05</td>
<td>3.48E-05</td>
</tr>
<tr>
<td>8</td>
<td>108</td>
<td>4.82E-05</td>
<td>4.82E-05</td>
<td>4.82E-05</td>
<td>4.82E-05</td>
</tr>
<tr>
<td>9</td>
<td>109</td>
<td>2.38E-05</td>
<td>2.38E-05</td>
<td>2.38E-05</td>
<td>2.38E-05</td>
</tr>
<tr>
<td>10</td>
<td>201</td>
<td>4.57E-05</td>
<td>4.57E-05</td>
<td>4.57E-05</td>
<td>4.57E-05</td>
</tr>
<tr>
<td>11</td>
<td>202</td>
<td>7.85E-05</td>
<td>7.85E-05</td>
<td>7.85E-05</td>
<td>7.85E-05</td>
</tr>
<tr>
<td>12</td>
<td>203</td>
<td>6.16E-05</td>
<td>8.37E-05</td>
<td>8.37E-05</td>
<td>8.37E-05</td>
</tr>
<tr>
<td>13</td>
<td>204</td>
<td>6.83E-05</td>
<td>6.83E-05</td>
<td>6.83E-05</td>
<td>6.83E-05</td>
</tr>
<tr>
<td>14</td>
<td>205</td>
<td>3.82E-05</td>
<td>3.82E-05</td>
<td>3.82E-05</td>
<td>3.82E-05</td>
</tr>
<tr>
<td>15</td>
<td>206</td>
<td>7.05E-05</td>
<td>7.05E-05</td>
<td>7.05E-05</td>
<td>7.05E-05</td>
</tr>
<tr>
<td>16</td>
<td>207</td>
<td>5.52E-05</td>
<td>5.52E-05</td>
<td>5.52E-05</td>
<td>5.52E-05</td>
</tr>
<tr>
<td>17</td>
<td>208</td>
<td>1.94E-05</td>
<td>1.94E-05</td>
<td>1.94E-05</td>
<td>1.94E-05</td>
</tr>
<tr>
<td>18</td>
<td>209</td>
<td>2.83E-07</td>
<td>2.83E-07</td>
<td>2.83E-07</td>
<td>2.83E-07</td>
</tr>
<tr>
<td>19</td>
<td>210</td>
<td>4.20E-05</td>
<td>4.20E-05</td>
<td>4.20E-05</td>
<td>4.20E-05</td>
</tr>
<tr>
<td>20</td>
<td>211</td>
<td>3.36E-05</td>
<td>3.36E-05</td>
<td>3.36E-05</td>
<td>8.64E-05</td>
</tr>
<tr>
<td>21</td>
<td>212</td>
<td>8.28E-05</td>
<td>4.52E-05</td>
<td>4.52E-05</td>
<td>8.28E-05</td>
</tr>
<tr>
<td>22</td>
<td>213</td>
<td>5.53E-05</td>
<td>5.53E-05</td>
<td>5.53E-05</td>
<td>5.53E-05</td>
</tr>
<tr>
<td>23</td>
<td>301</td>
<td>7.27E-05</td>
<td>7.27E-05</td>
<td>7.27E-05</td>
<td>7.27E-05</td>
</tr>
<tr>
<td>24</td>
<td>302</td>
<td>2.32E-05</td>
<td>9.18E-05</td>
<td>9.18E-05</td>
<td>2.32E-05</td>
</tr>
<tr>
<td>25</td>
<td>303</td>
<td>9.16E-05</td>
<td>9.16E-05</td>
<td>9.16E-05</td>
<td>9.16E-05</td>
</tr>
<tr>
<td>26</td>
<td>304</td>
<td>9.16E-05</td>
<td>9.16E-05</td>
<td>9.16E-05</td>
<td>9.16E-05</td>
</tr>
<tr>
<td>27</td>
<td>305</td>
<td>1.58E-06</td>
<td>-0.51831</td>
<td>-0.51831</td>
<td>1.58E-06</td>
</tr>
<tr>
<td>29</td>
<td>401</td>
<td>9.16E-05</td>
<td>0.001831</td>
<td>0.001831</td>
<td>0.001831</td>
</tr>
<tr>
<td>30</td>
<td>402</td>
<td>0.000101</td>
<td>0.000101</td>
<td>0.000101</td>
<td>0.000101</td>
</tr>
<tr>
<td>31</td>
<td>403</td>
<td>9.45E-05</td>
<td>0.000101</td>
<td>0.000101</td>
<td>0.000101</td>
</tr>
<tr>
<td>32</td>
<td>404</td>
<td>8.01E-05</td>
<td>0.000101</td>
<td>0.000101</td>
<td>0.000101</td>
</tr>
<tr>
<td>33</td>
<td>405</td>
<td>7.25E-05</td>
<td>9.54E-05</td>
<td>9.54E-05</td>
<td>9.54E-05</td>
</tr>
<tr>
<td>34</td>
<td>406</td>
<td>9.97E-05</td>
<td>0.000221</td>
<td>0.000221</td>
<td>0.000221</td>
</tr>
<tr>
<td>35</td>
<td>407</td>
<td>-9.95E-10</td>
<td>-9.95E-10</td>
<td>-9.95E-10</td>
<td>-9.95E-10</td>
</tr>
<tr>
<td>36</td>
<td>701</td>
<td>6.53E-05</td>
<td>28.187</td>
<td>17.0579</td>
<td>17.0579</td>
</tr>
<tr>
<td>37</td>
<td>702</td>
<td>5.21E-06</td>
<td>1.96278</td>
<td>1.96278</td>
<td>1.96278</td>
</tr>
<tr>
<td>38</td>
<td>703</td>
<td>-1.29E-07</td>
<td>-1.29E-07</td>
<td>-1.29E-07</td>
<td>-1.29E-07</td>
</tr>
</tbody>
</table>

Table 2.1 Detailed $f_{best}$ data results comparison in 4 different selection options
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Lower-y</th>
<th>Age</th>
<th>BoxSize</th>
<th>y-Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>101</td>
<td>0.016</td>
<td>0.031</td>
<td>0</td>
<td>0.031</td>
</tr>
<tr>
<td>2</td>
<td>102</td>
<td>0</td>
<td>0</td>
<td>0.015</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>103</td>
<td>0</td>
<td>0.015</td>
<td>0.016</td>
<td>0.016</td>
</tr>
<tr>
<td>4</td>
<td>104</td>
<td>0.015</td>
<td>0</td>
<td>0</td>
<td>0.031</td>
</tr>
<tr>
<td>5</td>
<td>105</td>
<td>0.016</td>
<td>0.016</td>
<td>0</td>
<td>0.016</td>
</tr>
<tr>
<td>6</td>
<td>106</td>
<td>0</td>
<td>0.015</td>
<td>0.015</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>107</td>
<td>0</td>
<td>0.015</td>
<td>0.031</td>
<td>0.046</td>
</tr>
<tr>
<td>8</td>
<td>108</td>
<td>0.016</td>
<td>0</td>
<td>0.016</td>
<td>0.031</td>
</tr>
<tr>
<td>9</td>
<td>109</td>
<td>0.016</td>
<td>0</td>
<td>0</td>
<td>0.031</td>
</tr>
<tr>
<td>10</td>
<td>201</td>
<td>0.015</td>
<td>0.047</td>
<td>0.047</td>
<td>0.032</td>
</tr>
<tr>
<td>11</td>
<td>202</td>
<td>0.093</td>
<td>0.14</td>
<td>0.203</td>
<td>0.125</td>
</tr>
<tr>
<td>12</td>
<td>203</td>
<td>2.512</td>
<td>6.131</td>
<td>10.405</td>
<td>5.74</td>
</tr>
<tr>
<td>13</td>
<td>204</td>
<td>0.063</td>
<td>0.047</td>
<td>0.078</td>
<td>0.062</td>
</tr>
<tr>
<td>14</td>
<td>205</td>
<td>0.016</td>
<td>0.016</td>
<td>0.031</td>
<td>0.016</td>
</tr>
<tr>
<td>15</td>
<td>206</td>
<td>0.016</td>
<td>0.015</td>
<td>0.016</td>
<td>0.031</td>
</tr>
<tr>
<td>16</td>
<td>207</td>
<td>0.405</td>
<td>0.468</td>
<td>0.968</td>
<td>0.717</td>
</tr>
<tr>
<td>17</td>
<td>208</td>
<td>0</td>
<td>0.046</td>
<td>0.063</td>
<td>0.031</td>
</tr>
<tr>
<td>18</td>
<td>209</td>
<td>0.031</td>
<td>0.047</td>
<td>0.046</td>
<td>0.031</td>
</tr>
<tr>
<td>19</td>
<td>210</td>
<td>0.062</td>
<td>0.047</td>
<td>0.062</td>
<td>0.078</td>
</tr>
<tr>
<td>20</td>
<td>211</td>
<td>0.032</td>
<td>0.047</td>
<td>0.078</td>
<td>0.078</td>
</tr>
<tr>
<td>22</td>
<td>213</td>
<td>0.156</td>
<td>0.421</td>
<td>0.609</td>
<td>0.577</td>
</tr>
<tr>
<td>23</td>
<td>301</td>
<td>0</td>
<td>0.047</td>
<td>0.063</td>
<td>0.047</td>
</tr>
<tr>
<td>24</td>
<td>302</td>
<td>0.28</td>
<td>0.561</td>
<td>0.843</td>
<td>0.546</td>
</tr>
<tr>
<td>25</td>
<td>303</td>
<td>0.016</td>
<td>0.016</td>
<td>0.032</td>
<td>0.031</td>
</tr>
<tr>
<td>26</td>
<td>304</td>
<td>1.341</td>
<td>0.827</td>
<td>1.685</td>
<td>1.092</td>
</tr>
<tr>
<td>27</td>
<td>305</td>
<td>0.047</td>
<td>0.78</td>
<td>1.154</td>
<td>0.125</td>
</tr>
<tr>
<td>28</td>
<td>306</td>
<td>0.031</td>
<td>0.062</td>
<td>0.078</td>
<td>0.032</td>
</tr>
<tr>
<td>29</td>
<td>401</td>
<td>0.031</td>
<td>1784.729</td>
<td>1804.245</td>
<td>1542.975</td>
</tr>
<tr>
<td>30</td>
<td>402</td>
<td>0</td>
<td>0.016</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>31</td>
<td>403</td>
<td>0.094</td>
<td>308.804</td>
<td>719.196</td>
<td>344.747</td>
</tr>
<tr>
<td>32</td>
<td>404</td>
<td>0.047</td>
<td>649.511</td>
<td>1585.438</td>
<td>647.903</td>
</tr>
<tr>
<td>33</td>
<td>405</td>
<td>0.141</td>
<td>1.201</td>
<td>1.95</td>
<td>1.404</td>
</tr>
<tr>
<td>34</td>
<td>406</td>
<td>30.28</td>
<td>621.149</td>
<td>1549.386</td>
<td>700.616</td>
</tr>
<tr>
<td>35</td>
<td>407</td>
<td>0.031</td>
<td>0.093</td>
<td>0.14</td>
<td>0.187</td>
</tr>
<tr>
<td>36</td>
<td>701</td>
<td>0.749</td>
<td>1811.858</td>
<td>15181.77</td>
<td>1577.544</td>
</tr>
<tr>
<td>37</td>
<td>702</td>
<td>0.078</td>
<td>1220.88</td>
<td>1803.496</td>
<td>1219.054</td>
</tr>
<tr>
<td>38</td>
<td>703</td>
<td>0.218</td>
<td>128.015</td>
<td>378.599</td>
<td>125.441</td>
</tr>
</tbody>
</table>

Table 2.2 Detailed CPU Time data results comparison in 4 different selection options
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Lower-y</th>
<th>Age</th>
<th>BoxSize</th>
<th>y-Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>101</td>
<td>19</td>
<td>21</td>
<td>21</td>
<td>23</td>
</tr>
<tr>
<td>2</td>
<td>102</td>
<td>13</td>
<td>19</td>
<td>20</td>
<td>18</td>
</tr>
<tr>
<td>3</td>
<td>103</td>
<td>16</td>
<td>26</td>
<td>26</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td>104</td>
<td>15</td>
<td>20</td>
<td>20</td>
<td>18</td>
</tr>
<tr>
<td>5</td>
<td>105</td>
<td>15</td>
<td>26</td>
<td>26</td>
<td>34</td>
</tr>
<tr>
<td>6</td>
<td>106</td>
<td>11</td>
<td>12</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>7</td>
<td>107</td>
<td>50</td>
<td>76</td>
<td>76</td>
<td>76</td>
</tr>
<tr>
<td>8</td>
<td>108</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>9</td>
<td>109</td>
<td>20</td>
<td>27</td>
<td>32</td>
<td>28</td>
</tr>
<tr>
<td>10</td>
<td>201</td>
<td>80</td>
<td>160</td>
<td>160</td>
<td>232</td>
</tr>
<tr>
<td>11</td>
<td>202</td>
<td>302</td>
<td>635</td>
<td>635</td>
<td>696</td>
</tr>
<tr>
<td>12</td>
<td>203</td>
<td>4884</td>
<td>10085</td>
<td>10085</td>
<td>9758</td>
</tr>
<tr>
<td>13</td>
<td>204</td>
<td>195</td>
<td>257</td>
<td>257</td>
<td>300</td>
</tr>
<tr>
<td>14</td>
<td>205</td>
<td>16</td>
<td>54</td>
<td>54</td>
<td>63</td>
</tr>
<tr>
<td>15</td>
<td>206</td>
<td>13</td>
<td>42</td>
<td>42</td>
<td>75</td>
</tr>
<tr>
<td>16</td>
<td>207</td>
<td>1328</td>
<td>2188</td>
<td>2188</td>
<td>2456</td>
</tr>
<tr>
<td>17</td>
<td>208</td>
<td>20</td>
<td>117</td>
<td>117</td>
<td>147</td>
</tr>
<tr>
<td>18</td>
<td>209</td>
<td>24</td>
<td>102</td>
<td>102</td>
<td>117</td>
</tr>
<tr>
<td>19</td>
<td>210</td>
<td>115</td>
<td>151</td>
<td>151</td>
<td>194</td>
</tr>
<tr>
<td>20</td>
<td>211</td>
<td>295</td>
<td>330</td>
<td>330</td>
<td>660</td>
</tr>
<tr>
<td>21</td>
<td>212</td>
<td>5629</td>
<td>12168</td>
<td>12168</td>
<td>11806</td>
</tr>
<tr>
<td>22</td>
<td>213</td>
<td>694</td>
<td>1837</td>
<td>1837</td>
<td>1763</td>
</tr>
<tr>
<td>23</td>
<td>301</td>
<td>24</td>
<td>112</td>
<td>112</td>
<td>165</td>
</tr>
<tr>
<td>24</td>
<td>302</td>
<td>1415</td>
<td>4099</td>
<td>4099</td>
<td>3430</td>
</tr>
<tr>
<td>25</td>
<td>303</td>
<td>20</td>
<td>32</td>
<td>32</td>
<td>46</td>
</tr>
<tr>
<td>26</td>
<td>304</td>
<td>3874</td>
<td>3567</td>
<td>3567</td>
<td>5443</td>
</tr>
<tr>
<td>27</td>
<td>305</td>
<td>108</td>
<td>4865</td>
<td>4865</td>
<td>445</td>
</tr>
<tr>
<td>28</td>
<td>306</td>
<td>66</td>
<td>333</td>
<td>333</td>
<td>294</td>
</tr>
<tr>
<td>29</td>
<td>401</td>
<td>86</td>
<td>150000</td>
<td>98520</td>
<td>150000</td>
</tr>
<tr>
<td>30</td>
<td>402</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>31</td>
<td>403</td>
<td>416</td>
<td>106559</td>
<td>106559</td>
<td>99083</td>
</tr>
<tr>
<td>32</td>
<td>404</td>
<td>93</td>
<td>119988</td>
<td>119988</td>
<td>119964</td>
</tr>
<tr>
<td>33</td>
<td>405</td>
<td>750</td>
<td>6909</td>
<td>6909</td>
<td>6816</td>
</tr>
<tr>
<td>34</td>
<td>406</td>
<td>22376</td>
<td>150000</td>
<td>150000</td>
<td>150000</td>
</tr>
<tr>
<td>35</td>
<td>407</td>
<td>62</td>
<td>692</td>
<td>665</td>
<td>687</td>
</tr>
<tr>
<td>36</td>
<td>701</td>
<td>2444</td>
<td>89060</td>
<td>150000</td>
<td>150000</td>
</tr>
<tr>
<td>37</td>
<td>702</td>
<td>120</td>
<td>150000</td>
<td>103020</td>
<td>150000</td>
</tr>
<tr>
<td>38</td>
<td>703</td>
<td>572</td>
<td>131524</td>
<td>131524</td>
<td>117180</td>
</tr>
</tbody>
</table>

Table 2.3: Detailed iteration data results comparison in 4 different selection options
For step 6, choose a coordinate direction for the splitting of the new box. For this procedure there are two approaches as follows:

1. Choose the direction with the longest side.
2. Choose the most influential direction. This case will be discussed in more detail in Chapter 3, where binary tree data structure and numerical results will be presented.

2.4 Standard Constraint Treatment Methods

The Hansen algorithm is normally used for solving unconstrained global optimization problems when the sampling is done at midpoints. While in [1] the constrained version algorithm was introduced, it doesn’t provide numerical results to support its effectiveness. The Hansen algorithm can be extended to solve constrained problems (C-GOP) by additional testing of a sample point’s feasibility and deletions based on constraint functions. Many specific strategies are presented in [1]. The constrained version of Hansen’s algorithm can be briefly stated as follows:

1. Input data: the dimension of the problem, \( n \); the domain \( X \) of the problem; inclusion functions \( F, H, G \) of \( f, h, g \) respectively.
2. Set \( Y := X, y := \text{lb } F(y), c = 0. \)
3. If a feasible point \( \bar{x} \) is given, set \( \bar{f} := \text{ub } F(\bar{x}) \), else set \( \bar{f} := \infty \).
4. Initialize the list \( L = \{(Y, y, c)\} \).
5. Choose a box from the list \( L \), and remove it from the list.
6. Choose a coordinate direction for the splitting of the new \( Y \).
7. Bisect \( Y \) normal to the chosen direction, getting the boxes \( V_1, V_2 \), with corresponding lower bounds \( y_1, y_2 \) and flags \( c_1 \) and \( c_2 \).
8. For $i = 1, 2$
   a. Delete $(V_i, y_i, c_i)$ if it can be proven that $V_i$ contains no optimal solution (e.g. when $\bar{f} < y_i$, or one of the constraints is totally violated);
   b. Sampling at the midpoint and possible update of $\bar{f}$;
   c. Store $(V_i, y_i, c_i)$ into the list if it is not deleted.
9. If one of the termination criteria hold, stop. Otherwise, go to Step 5.

It is a standard algorithm for C-GOP. There are many methods discussed in [1] that can be used to solve the constrained problem, such as Kuhn-Tucker method, penalty method, exhausting method, and statistical method. The Kuhn-Tucker method is used to solve the Kuhn-Tucker conditions in order to obtain the local minimizers which include the global minimizers [1]. The penalty method is used to reduce the constrained case to a sequence of unconstrained problems [1]. Exhaustion method is used to check the whole area under consideration piecewise for a certain property [1]. Therefore, the area can be reduced to a smaller one or eliminating the pieces with the unwanted property [1]. Statistical method is very effective, but reliability cannot be guaranteed [1]. There are also several papers dealing with constraint problems recently. Some of them are summarized below.

In [14] [17], [18] and [23], they demonstrated the Exclusion test in the area of interval analysis for finding solutions of nonlinear system of equations over a compact domain. Furthermore, F. Kalovics [22] constructed a zone to include a subinterval for a specific function and created zone functions. Adding zone functions can help reduce unnecessary infeasible regions and it helps in the deletion step. In [21] and [22], the improvement of the Tree search efficiency for constraint satisfaction problems is considered. In this paper, two principles, i.e.
back tracking search and forward checking search are introduced. We can apply this approach in bounding constraints in order to reduce each variable’s intervals.

Besides the interval branch and bound method discussed in the introduction section, there is another approach to solve the C-GOP and it is called local sampling strategies. In [52], Min Sun and A.W. Johnson proposed three different sampling procedures including: 1) local optimization search sampling; 2) random sampling; 3) global fake sampling. This approach can be used to get a feasible solution other than those approaches we mentioned above.

In this dissertation, we apply these strategies discussed above. We combine the interval branch and bound method and the local sampling strategies in order to get a better optimal solution faster. We use the tree methods to update all functions’ information, including upper bounds, lower bounds, and each node’s ID. Detailed ID information will be discussed in section 3.3. We have added Affine Arithmetic as another approach to get the inclusion function, and it can be used to sharpen the error approximation conveniently. Numerical results will be shown in each section.

Moreover, we extend the idea mentioned in [52] which is to find the local sampling point and develop new methodologies to deal with both one linear constraint as well as two linear constraints.

For one linear constraint, we develop a backward iterative interval detecting method. First, we initialize all $x_i$’s values to be their lower bounds except the last one $x_n$. Second, for the last variable $x_n$, we look at its domain to check whether we can find a value such that $\text{lb}(\sum_{i=1}^{n-1} c_i x_i) + x_n$ is equal to the right hand side value. If not, one reasonable value is chosen, and then the process proceeds to $x_{n-1}$ to get a proper value. By repeating these steps we can ultimately find a sampling point.
For two linear constraints, we develop three methods: a traditional coordinate descent method, a cutting line method, and a projection coordinate descent method. The latter two methods’ idea was inspired by the Affine Arithmetic in [3] with the usage of the noise symbol. In [3], it said that a quantity $x$ can be represented by an expression in the form

$$
\hat{x} = x_0 + x_1\varepsilon_1 + x_2\varepsilon_2 + \cdots + x_n\varepsilon_n; \quad (2.4.1)
$$

where $\varepsilon_i$ is the noise symbol. This form has the property that for each $\varepsilon_i$ it is within the interval $[-1, 1]$. In real practice, the smaller the dimension, the easier it is to find a solution. Since we are dealing with multiple dimensions, finding a solution will be easier if they can be transformed to fewer dimensions. We set both equations’ right hand side values $a$ and $b$ to be the ordered pair $(a, b)$ in a new 2-dimensional $\alpha - \beta$ coordinate system. Then we treat all variables as the noise symbols and normalize them into $[-1,1]$ to build a new system of equations. This step is beneficial and can help us to locate a solution more easily. After the new system of equations is built, we project each variable in 2 parallel straight lines in the $\alpha - \beta$ coordinate system, which may enclose a feasible bounded region. We extend two different methods in order to satisfy the feasibility. First, we parallelize each variable’s bounded line to go through the feasible point $(a, b)$ so that two intersection points are obtained. Then we project each variable’s value based on the location of $(a, b)$. Last we apply the coordinate descent method to approach the point $(a, b)$. The convergence is not very bad since the projection point is already quite close. The second method is the cutting line method. In this method, we use one variable’s parallel line to obtain the feasible point $(a, b)$. If the line can go through this point within the variable’s normalized domain $[-1, 1]$, we get the solution. If not, we change it to another variable. Eventually, we will find a solution since the projection field is guaranteed to contain at least one feasible solution. Because we can find the feasible solution in a finite number steps (no more than the variable’s
dimensions), the convergence for the cutting line method is good.

However, most of the examples demonstrated in the traditional IA methods seem to be applied in limited dimensional problems (much less than 100 dimensions) [52]. There are two major concerns in solving large dimensional problems: large amount of memory space and slow speed of convergence [52]. The number of sub-boxes to be processed in an interval method could potentially increase exponentially when the dimension of the domain increases [52]. Inspired by such observations, we have developed some new strategies associated with the interval branch and bound methodology [52]. Our new versions of the interval algorithm including the projection coordinate descent method, the coordinate descent method, and the cutting line method in step 8 in this dissertation show improvement both in reducing memory space usage and speeding convergence [52]. Detailed data and graphs will be provided to better demonstrate the idea. All the data will be compared with the standard Hansen method.
CHAPTER 3

IMPROVEMENT STRATEGIES FOR INTERVAL METHODS IN GLOBAL OPTIMIZATION

3.1 Binary Tree Data Structure

This is a method for selecting the subdivision directions which causes appropriate reduction in child boxes. This idea is inspired by [13] and [20]. [4], [21] and [22] provide us a specified method to utilize the tree data structure. Binary Tree Data Structure focuses on how to trace boxes and record the sequence of variables splitting after a given box. This method can identify major impact variables in a constraint expression, and thus it guides the algorithm in identifying the best variable to be partitioned, which is called “the most influential direction” in this dissertation. This method is effective in reducing CPU time, and numerical results will be provided at the end of this section.

A binary tree representing the objective function is built as follows. Leaves of the binary tree are elements obtained from the function; they are either variables or constants. All other nodes represent binary expressions of the form (left Θ right). A binary operator “Θ” is an arithmetic operator (×, ÷, +, −) called a parent box having two branches or two child boxes (“left”, “right”) that are themselves recursive binary sub-trees. However, mathematical functions such as ln, exp, sin, cos, tan, etc, are unary operators which only have one branch or one child. In such cases, the argument of the function is always placed in the “left” branch or child.
The first iterative explores all nodes generated at a given tree level before it starts assessing the nodes’ detailed information at the next stage. Exploration of boxes at the same stage can be done in any order, it may start from best-first box or the one on the most right or most left at that stage. In our algorithm, for convenience, we choose the most left one of the current stage. On the other hand, in the proposed tree data structure, a node which is called parent box is permitted to grow a sub-tree existing of its children or child if it only has one child. Thus it forms a partial tree structure. Based on these children’s information, we are able to determine which one plays the more important role according different rules such as comparing the absolute values or the width of the interval for example. After that, a child box is selected among the children of the same parent. We do the same procedures repeatedly until we reach the root of the tree.

This technique can be applied in different situations. For example, we can apply it in finding the objective function’s minimum, or we can place it in the constraint to delete the child box. Based on these situations, we also separate it into three cases, which will be illustrated in a detailed example in the following situation. We have added this feature in the selection step which is step 2 in Hansen’s Algorithm.

**Example 3.1:** Given the domain is \([-1, 1.5] \times [-1, 1.5]\), the objective function (example 201 in the appendix) is

\[
 f(x_1, x_2) = [x_1^2 - 0.3 \cos(3\pi x_1)] + [2x_2^2 - 0.4 \cos(4\pi x_2)] + 0.7.
\]

The detailed binary tree graph for this example is as follows:
Figure 3.1 Binary Tree Data Structure for Example 201

Tracing the lower bound, normally used to find the minimum of the objective function value.
The following table shows the effectiveness of the binary tree data structure. We compared the minimum objective function value $f_{\text{best}}$, CPU time (ms), and iteration numbers with results obtained by applying the standard Hansen method. Here is the comparison:

<table>
<thead>
<tr>
<th>Ex201</th>
<th>Standard Hansen</th>
<th>Binary Tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_{\text{best}}$</td>
<td>$4.56869 \times 10^{-5}$</td>
<td>$3.2026 \times 10^{-5}$</td>
</tr>
<tr>
<td>CPU(ms)</td>
<td>31</td>
<td>$&lt;1$</td>
</tr>
<tr>
<td>Iterations</td>
<td>80</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 3.1 Comparison for example 201

From the above table, we can see that by applying the binary tree data structure, all data including $f_{\text{best}}$, CPU time, and iteration numbers are improved compared with the standard Hansen method by selecting the maximum width of the box in the list. Thus we conclude that
choosing a most influential direction in the binary tree data structure is an effective feature to accelerate the calculation. We applied this feature to 31 other examples, for which we have the comparison graphs as follows:

Remark: In the following three graphs, we used the data generated from the standard Hansen method to subtract the data generated from applying choosing the most important direction method in binary tree data structure (called H-Dir). Thus if the data is below the x-axis, it means the H-Dir is better.

![Figure 3.3](image)

Figure 3.3  $f_{\text{best}}$’s difference between the Hansen and the H-Dir
Figure 3.4  CPU Time difference between the Hansen and the H-Dir

Figure 3.5 Iteration difference between the Hansen and the H-Dir
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Hansen</th>
<th>H-Dir</th>
<th>D</th>
<th>Hansen</th>
<th>H-Dir</th>
<th>D</th>
<th>Hansen</th>
<th>H-Dir</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>201</td>
<td>4.57E-05</td>
<td>3.20E-05</td>
<td>-1.37E-05</td>
<td>15</td>
<td>0</td>
<td>-15</td>
<td>80</td>
<td>20</td>
<td>-60</td>
</tr>
<tr>
<td>2</td>
<td>202</td>
<td>7.85E-05</td>
<td>7.85E-05</td>
<td>0</td>
<td>93</td>
<td>47</td>
<td>-46</td>
<td>302</td>
<td>279</td>
<td>-23</td>
</tr>
<tr>
<td>3</td>
<td>203</td>
<td>6.16E-05</td>
<td>6.16E-05</td>
<td>0</td>
<td>2512</td>
<td>2387</td>
<td>-125</td>
<td>4884</td>
<td>4764</td>
<td>-120</td>
</tr>
<tr>
<td>4</td>
<td>204</td>
<td>6.83E-05</td>
<td>6.83E-05</td>
<td>0</td>
<td>63</td>
<td>63</td>
<td>0</td>
<td>195</td>
<td>195</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>205</td>
<td>3.82E-05</td>
<td>3.82E-05</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>-16</td>
<td>16</td>
<td>15</td>
<td>-1</td>
</tr>
<tr>
<td>6</td>
<td>206</td>
<td>7.05E-05</td>
<td>7.05E-05</td>
<td>0</td>
<td>16</td>
<td>16</td>
<td>0</td>
<td>13</td>
<td>12</td>
<td>-1</td>
</tr>
<tr>
<td>7</td>
<td>207</td>
<td>5.52E-05</td>
<td>5.52E-05</td>
<td>0</td>
<td>405</td>
<td>312</td>
<td>-93</td>
<td>1328</td>
<td>1128</td>
<td>-200</td>
</tr>
<tr>
<td>8</td>
<td>208</td>
<td>1.94E-05</td>
<td>1.94E-05</td>
<td>0</td>
<td>32</td>
<td>32</td>
<td>0</td>
<td>295</td>
<td>295</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>209</td>
<td>2.83E-07</td>
<td>2.83E-07</td>
<td>0</td>
<td>31</td>
<td>0</td>
<td>-31</td>
<td>24</td>
<td>12</td>
<td>-12</td>
</tr>
<tr>
<td>10</td>
<td>210</td>
<td>4.20E-05</td>
<td>4.20E-05</td>
<td>0</td>
<td>62</td>
<td>31</td>
<td>-31</td>
<td>115</td>
<td>110</td>
<td>-5</td>
</tr>
<tr>
<td>11</td>
<td>211</td>
<td>3.36E-05</td>
<td>3.36E-05</td>
<td>0</td>
<td>32</td>
<td>32</td>
<td>0</td>
<td>295</td>
<td>295</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>212</td>
<td>8.28E-05</td>
<td>8.28E-05</td>
<td>0</td>
<td>4633</td>
<td>4458</td>
<td>-175</td>
<td>5629</td>
<td>5428</td>
<td>-201</td>
</tr>
<tr>
<td>13</td>
<td>213</td>
<td>5.53E-05</td>
<td>5.53E-05</td>
<td>0</td>
<td>156</td>
<td>140</td>
<td>-16</td>
<td>694</td>
<td>694</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>301</td>
<td>7.27E-05</td>
<td>7.27E-05</td>
<td>0</td>
<td>0</td>
<td>32</td>
<td>32</td>
<td>24</td>
<td>24</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>302</td>
<td>2.32E-05</td>
<td>2.32E-05</td>
<td>0</td>
<td>280</td>
<td>250</td>
<td>-30</td>
<td>1415</td>
<td>1394</td>
<td>-21</td>
</tr>
<tr>
<td>16</td>
<td>303</td>
<td>9.16E-05</td>
<td>9.16E-05</td>
<td>0</td>
<td>16</td>
<td>15</td>
<td>-1</td>
<td>20</td>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>304</td>
<td>9.16E-05</td>
<td>9.16E-05</td>
<td>0</td>
<td>1341</td>
<td>1295</td>
<td>-46</td>
<td>3874</td>
<td>3820</td>
<td>-54</td>
</tr>
<tr>
<td>18</td>
<td>305</td>
<td>1.58E-06</td>
<td>1.58E-06</td>
<td>0</td>
<td>47</td>
<td>62</td>
<td>15</td>
<td>108</td>
<td>108</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>306</td>
<td>-3.09E-05</td>
<td>-3.09E-05</td>
<td>0</td>
<td>31</td>
<td>30</td>
<td>-1</td>
<td>66</td>
<td>61</td>
<td>-5</td>
</tr>
<tr>
<td>20</td>
<td>401</td>
<td>9.16E-05</td>
<td>9.16E-05</td>
<td>0</td>
<td>31</td>
<td>30</td>
<td>-1</td>
<td>86</td>
<td>80</td>
<td>-6</td>
</tr>
<tr>
<td>21</td>
<td>402</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>22</td>
<td>403</td>
<td>9.45E-05</td>
<td>9.45E-05</td>
<td>0</td>
<td>94</td>
<td>125</td>
<td>31</td>
<td>416</td>
<td>416</td>
<td>0</td>
</tr>
<tr>
<td>23</td>
<td>404</td>
<td>8.01E-05</td>
<td>8.01E-05</td>
<td>0</td>
<td>47</td>
<td>63</td>
<td>16</td>
<td>93</td>
<td>93</td>
<td>0</td>
</tr>
<tr>
<td>24</td>
<td>405</td>
<td>7.25E-05</td>
<td>7.25E-05</td>
<td>0</td>
<td>141</td>
<td>156</td>
<td>15</td>
<td>750</td>
<td>750</td>
<td>0</td>
</tr>
<tr>
<td>25</td>
<td>406</td>
<td>9.97E-05</td>
<td>9.97E-05</td>
<td>0</td>
<td>30280</td>
<td>29937</td>
<td>-343</td>
<td>22376</td>
<td>21961</td>
<td>-415</td>
</tr>
<tr>
<td>26</td>
<td>407</td>
<td>-9.95E-10</td>
<td>-9.95E-10</td>
<td>0</td>
<td>31</td>
<td>32</td>
<td>1</td>
<td>62</td>
<td>62</td>
<td>0</td>
</tr>
<tr>
<td>27</td>
<td>701</td>
<td>6.53E-05</td>
<td>6.53E-05</td>
<td>0</td>
<td>749</td>
<td>702</td>
<td>-47</td>
<td>2444</td>
<td>2421</td>
<td>-23</td>
</tr>
<tr>
<td>28</td>
<td>702</td>
<td>5.21E-06</td>
<td>5.21E-06</td>
<td>0</td>
<td>78</td>
<td>78</td>
<td>0</td>
<td>120</td>
<td>120</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>703</td>
<td>-1.29E-07</td>
<td>-1.29E-07</td>
<td>0</td>
<td>218</td>
<td>188</td>
<td>-30</td>
<td>572</td>
<td>542</td>
<td>-30</td>
</tr>
</tbody>
</table>

Table 3.2 Detailed data results between the Hansen and the H-Dir

From three graphs and the table above, we can see that although the most important direction feature is used, the optimal objective function values are almost the same. When we look at the CPU time and the number of iteration from the above table (the negative numbers mean the H-Dir is better), both of them are improved. For some cases, both CPU time and the number of
iteration are improved a lot. Hence we conclude that applying the Binary Tree Data structure is more effective.

3.2 Affine Arithmetic

Affine Arithmetic method is derived from inner arithmetic [25]. There are many papers published recently discussing this method, such as in [3, 6, 19]. As we mentioned above, Inclusion function \( F(X) \) encloses the true range of \( f(x) \). In this case, the interval obtained by interval arithmetic may be much wider than the exact range.

Here are two examples that can show the error approximation we discussed above.

**Example 3.2:**

a. The given objective function is \( f(x) = x \cdot x, X = [-1, 1] \)

In this case, \( x \cdot x = x^2 \geq 0 \), \( f(X) \) should be \([0,1]\). However, after applying IA multiplication formula, we got \( F(X) = [-1, 1] \) which doubles the diameter of \([0, 1]\).

b. \( f(x) = x^2 - x^2, x \in [0,1], f(X) \) should be 0 since it is a function minus itself.

However, the interval subtraction formula gives \( F(X) = [-1, 1] \) whose diameter is twice the diameter of \([0, 1]\), instead of zero.

One of the main concerns when we use interval methods is how to reduce the error approximation generated with standard interval arithmetic (IA), especially if we have confronted a complicated problem or long computation.

Affine arithmetic (AA) [3] is a more compound and expensive computation model, designed to give tighter and more informative bounds than IA in certain situations. Moreover,
AA provides a linear representation for the joint range of related magnitudes that can be generated to raise the productivity of IA.

In Affine Arithmetic a mass \( x \) is denoted by the form

\[
\hat{x} = x_0 + x_1 \varepsilon_1 + x_2 \varepsilon_2 + \cdots + x_n \varepsilon_n;
\]  

which is an affine expression on the noise symbols \( \varepsilon_i \) with floating point coefficient \( x_i \) [3]. Each noise symbol \( \varepsilon_i \) is a symbolic real variable whose value is unknown except that it is regulated to the domain \( U = [-1,1] \) and is independent from the other noise symbols [3]. The coefficient \( x_0 \) is called the central value of the affine form \( \hat{x} \) [3]. The coefficients \( x_1, \ldots, x_n \) are called the partial deviations linked with the noise symbols \( \varepsilon_1, \ldots, \varepsilon_n \) in \( \hat{x} \) [3]. The number \( n \) of noise symbols depends on the affine form: different affine forms can use a different number of noise symbols, some of which may be shared with other affine forms [3].

This feature helps in “Bounding” step which is step 2 in finding the inclusion function for \( F, H, \) and \( G \). We applied this method in some unconstrained problems, and here are 3 graphs about the comparison between the standard Hansen method and the Affine Arithmetic method in optimal objective function values, CPU time (s), and number of iterations.

Remark: for all the following graphs, we all use the data generated from the standard Hansen method to minus the data generated from the Affine Arithmetic method. Therefore, if the line showed in the graphs is below the x-axis, it means the AA method is better.
Figure 3.6 $f_{\text{best}}$’s difference between the Hansen and the AA method

Figure 3.7 CPU time difference between the Hansen and the AA method
Figure 3.8 Iteration difference between the Hansen and the AA method
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Hansen</th>
<th>AA</th>
<th>D</th>
<th>Hansen</th>
<th>AA</th>
<th>D</th>
<th>iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>101</td>
<td>6.67E-05</td>
<td>6.67E-05</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>-16</td>
<td>19</td>
</tr>
<tr>
<td>2</td>
<td>102</td>
<td>1.10E-06</td>
<td>1.10E-06</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>16</td>
<td>13</td>
</tr>
<tr>
<td>3</td>
<td>103</td>
<td>1.32E-06</td>
<td>1.32E-06</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>4</td>
<td>104</td>
<td>2.87E-05</td>
<td>2.87E-05</td>
<td>0</td>
<td>15</td>
<td>0</td>
<td>-15</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>105</td>
<td>5.38E-05</td>
<td>5.38E-05</td>
<td>0</td>
<td>16</td>
<td>16</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>6</td>
<td>106</td>
<td>3.48E-05</td>
<td>3.48E-05</td>
<td>0</td>
<td>0</td>
<td>31</td>
<td>31</td>
<td>11</td>
</tr>
<tr>
<td>7</td>
<td>107</td>
<td>9.40E-05</td>
<td>9.40E-05</td>
<td>0</td>
<td>0</td>
<td>32</td>
<td>32</td>
<td>50</td>
</tr>
<tr>
<td>8</td>
<td>108</td>
<td>4.82E-05</td>
<td>4.82E-05</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>-16</td>
<td>12</td>
</tr>
<tr>
<td>9</td>
<td>109</td>
<td>2.38E-05</td>
<td>2.38E-05</td>
<td>0</td>
<td>16</td>
<td>15</td>
<td>-1</td>
<td>20</td>
</tr>
<tr>
<td>10</td>
<td>201</td>
<td>4.57E-05</td>
<td>4.57E-05</td>
<td>0</td>
<td>15</td>
<td>16</td>
<td>1</td>
<td>80</td>
</tr>
<tr>
<td>11</td>
<td>202</td>
<td>7.85E-05</td>
<td>7.85E-05</td>
<td>0</td>
<td>93</td>
<td>63</td>
<td>-30</td>
<td>302</td>
</tr>
<tr>
<td>12</td>
<td>203</td>
<td>6.16E-05</td>
<td>8.37E-05</td>
<td>2.21E-05</td>
<td>2512</td>
<td>93</td>
<td>2419</td>
<td>4884</td>
</tr>
<tr>
<td>13</td>
<td>204</td>
<td>6.83E-05</td>
<td>6.83E-05</td>
<td>0</td>
<td>63</td>
<td>172</td>
<td>109</td>
<td>195</td>
</tr>
<tr>
<td>14</td>
<td>205</td>
<td>3.82E-05</td>
<td>3.82E-05</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>-16</td>
<td>16</td>
</tr>
<tr>
<td>15</td>
<td>206</td>
<td>7.05E-05</td>
<td>7.05E-05</td>
<td>0</td>
<td>16</td>
<td>31</td>
<td>15</td>
<td>13</td>
</tr>
<tr>
<td>16</td>
<td>207</td>
<td>5.52E-05</td>
<td>5.52E-05</td>
<td>0</td>
<td>405</td>
<td>78</td>
<td>-327</td>
<td>1328</td>
</tr>
<tr>
<td>17</td>
<td>208</td>
<td>1.94E-05</td>
<td>1.94E-05</td>
<td>0</td>
<td>16</td>
<td>31</td>
<td>31</td>
<td>20</td>
</tr>
<tr>
<td>18</td>
<td>209</td>
<td>2.83E-07</td>
<td>2.83E-07</td>
<td>0</td>
<td>31</td>
<td>16</td>
<td>-15</td>
<td>24</td>
</tr>
<tr>
<td>20</td>
<td>211</td>
<td>3.36E-05</td>
<td>3.36E-05</td>
<td>0</td>
<td>32</td>
<td>94</td>
<td>62</td>
<td>295</td>
</tr>
<tr>
<td>21</td>
<td>212</td>
<td>8.28E-05</td>
<td>8.28E-05</td>
<td>0</td>
<td>4633</td>
<td>1014</td>
<td>3619</td>
<td>5629</td>
</tr>
<tr>
<td>22</td>
<td>213</td>
<td>5.53E-05</td>
<td>5.53E-05</td>
<td>0</td>
<td>156</td>
<td>171</td>
<td>15</td>
<td>694</td>
</tr>
<tr>
<td>23</td>
<td>301</td>
<td>7.27E-05</td>
<td>7.27E-05</td>
<td>0</td>
<td>0</td>
<td>47</td>
<td>47</td>
<td>24</td>
</tr>
<tr>
<td>24</td>
<td>401</td>
<td>9.16E-05</td>
<td>9.16E-05</td>
<td>0</td>
<td>31</td>
<td>31</td>
<td>0</td>
<td>86</td>
</tr>
<tr>
<td>30</td>
<td>402</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>31</td>
<td>403</td>
<td>9.45E-05</td>
<td>9.45E-05</td>
<td>0</td>
<td>94</td>
<td>63</td>
<td>-31</td>
<td>416</td>
</tr>
<tr>
<td>32</td>
<td>404</td>
<td>8.01E-05</td>
<td>8.01E-05</td>
<td>-4E-10</td>
<td>47</td>
<td>920</td>
<td>873</td>
<td>93</td>
</tr>
<tr>
<td>33</td>
<td>405</td>
<td>7.25E-05</td>
<td>7.25E-05</td>
<td>0</td>
<td>141</td>
<td>31</td>
<td>-10</td>
<td>750</td>
</tr>
<tr>
<td>34</td>
<td>406</td>
<td>9.97E-05</td>
<td>9.97E-05</td>
<td>0</td>
<td>30280</td>
<td>78</td>
<td>-30202</td>
<td>22376</td>
</tr>
<tr>
<td>35</td>
<td>407</td>
<td>-9.95E-10</td>
<td>-9.95E-10</td>
<td>0</td>
<td>31</td>
<td>78</td>
<td>47</td>
<td>62</td>
</tr>
<tr>
<td>36</td>
<td>701</td>
<td>6.53E-05</td>
<td>6.53E-05</td>
<td>0</td>
<td>749</td>
<td>125</td>
<td>-624</td>
<td>2444</td>
</tr>
<tr>
<td>37</td>
<td>702</td>
<td>5.21E-06</td>
<td>5.21E-06</td>
<td>0</td>
<td>78</td>
<td>78</td>
<td>0</td>
<td>120</td>
</tr>
<tr>
<td>38</td>
<td>703</td>
<td>-1.29E-07</td>
<td>-1.29E-07</td>
<td>0</td>
<td>218</td>
<td>125</td>
<td>-93</td>
<td>572</td>
</tr>
</tbody>
</table>

Table 3.3 Detailed data results between the Hansen and the AA method
From above three figures, although the $f_{\text{best}}$ is slightly bigger in the AA method, it is under
tolerance ($< 0.0000611$), so it can be ignored. When we looked at CPU time and number of
iterations, the graphs and data table indicate that both the CPU time, and the number of iterations
are better. For some cases, they are significantly better. Thus, we can conclude the Affine
Arithmetic method works effectively.

3.3 Exclusion Zone Functions

There are also some papers [5, 22, 24, 26] introduced that Exclusion Zone test is a useful
tool to be applied in IA. The most vital property of continuous multivariate real functions, which
is comparable to the property performing in the definition of continuity, is as follows.

Definition 3.1[22]: Let $D \subset \mathbb{R}^m$ be a finite open box, $p: D \rightarrow \mathbb{R}$ be a continuous function
on $D$, $c$ is a point of $D$, $\alpha$ is a real number $p(\circ) \neq \alpha$, then there is a neighborhood(zone, nonempty
open box) around the point $c$ in which the function value is not equal to $\alpha$ anywhere.

Definition 3.2 [22]: The function

$$Z_p: D \times \mathbb{R} \rightarrow \mathbb{R}^m, (c, \alpha) \rightarrow Z_p(c, \alpha). \quad (3.3.1)$$

is called a zone function of the function $p$, if $x$ is an element of the open box $Z_p(c, \alpha) \subseteq D$ and
$x \in Z_p(c, \alpha)$ implies $p(x) \neq \alpha$, for all $c \in D, p(x) \neq \alpha \in \mathbb{R}$; furthermore $Z_p(c, \alpha)$ is the empty
set if $p(x) = \alpha$.

Facts. If $p: D \subset \mathcal{R}^m \rightarrow \mathcal{R}$ and $q: D \subset \mathcal{R}^m \rightarrow \mathcal{R}$ are continuous functions on the open box $D$;
$Z_p, Z_q$ are their zone functions, $e$ is one of the allowed elementary functions, furthermore, $p/q$
and $e \circ q$ are well defined on $D$, then the functions
\[ Z_{p+\lambda}(c, \alpha) = Z_p(c, \beta), \lambda \in R, \beta = \alpha - \lambda; \]

\[ Z_{\lambda p}(c, \alpha) = Z_p(c, \beta), 0 \neq \lambda \in R, \beta = \alpha/\lambda; \]

\[ Z_{p+q}(c, \alpha) = \begin{cases} 
Z_p(c, \beta) \cap Z_q(c, \gamma), & \beta = \frac{\alpha + p(c) - q(c)}{2}, \gamma = \alpha - \beta. 
\end{cases} \quad (3.3.2) \]

\[ Z_{p/q}(c, \alpha) = \begin{cases} 
Z_p(c, \beta) \cap Z_q(c, \gamma), & \gamma = \frac{\alpha p(c) + q(c)}{1 + \alpha^2}, \beta = \alpha \gamma, 
\end{cases} \]

\[ Z_{e_{pq}}(c, \alpha) = \begin{cases} 
D \cap Z_q(c, \beta^c) \cap Z_q(c, \beta^c), & \beta^c \text{ is the maximum value for which } e(\beta^c) = \alpha, \beta^c < q(c), 
\end{cases} \]

\[ \beta^c \text{ is the minimum value for which } e(\beta^c) = \alpha, \beta^c > q(c), \]

are zone functions respectively.

Here we give the expressions with its coded ID:

<table>
<thead>
<tr>
<th>( l )</th>
<th>( x + \lambda )</th>
<th>( \lambda x )</th>
<th>( x + y )</th>
<th>( \frac{x}{y} )</th>
<th>( xy )</th>
<th>( x^\lambda )</th>
<th>( \lambda^x )</th>
</tr>
</thead>
<tbody>
<tr>
<td>8:</td>
<td>( \log x^\lambda )</td>
<td>9:</td>
<td>(</td>
<td>x</td>
<td>)</td>
<td>10:</td>
<td>( \sin x )</td>
</tr>
<tr>
<td>12:</td>
<td>( \tan x )</td>
<td>13:</td>
<td>( \cot x )</td>
<td>14:</td>
<td>( \arcsin x )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15:</td>
<td>( \arccos x )</td>
<td>16:</td>
<td>( \arctan x )</td>
<td>17:</td>
<td>( \arccot x )</td>
<td>18:</td>
<td>( \sinh x )</td>
</tr>
<tr>
<td>19:</td>
<td>( \cosh x )</td>
<td>20:</td>
<td>( \tanh x )</td>
<td>21:</td>
<td>( \coth x )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22:</td>
<td>( \arcsinh x )</td>
<td>23:</td>
<td>( \arccosh x )</td>
<td>24:</td>
<td>( \arctanh x )</td>
<td>25:</td>
<td>( \arccoth x )</td>
</tr>
<tr>
<td>26:</td>
<td>( \frac{x - \lambda}{\lambda - x} )</td>
<td>27:</td>
<td>( \frac{x - \lambda}{\lambda - x} )</td>
<td>28:</td>
<td>( x - y )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.4 Zone functions and their coded IDs

where \( \lambda \) is an integer or real constant. We added three IDs (26, 27, 28) in addition to [22]. We use the code form \((\text{func}, \text{pos}_x, \text{pos}_y \text{ or } \lambda \text{ or } 0)\) from [22], where \( 1 \leq \text{func} \leq 28 \) gives the serial number of the chosen function, the nonzero integer \( \text{pos}_x \) gives the place of the coordinate of a
or $b$ used as $x$, and the nonzero integer $\text{pos}_y$ gives the place of the coordinate of $a$ or $b$ used as $y$ [22].

**Example 3.3:**

$$(1, -1, 10.2) = a_1 + 10.2,$$

$$(2, 6, 10) = b_2^{10}.$$ 

Therefore, the objective function and constraints can be described by a sequence of the introduced code. This method helps in the “Reduction” step; it can help reduce more intervals or “boxes”. Here are 3 graphs about the comparison between the standard Hansen method and the Exclusion Zone method in optimal objective function values, CPU time (s), and number of iterations.

Remark: In the following three graphs, we used the data generated from the standard Hansen method to subtract by the data generated from Exclusion Zone method. Therefore, if the data is below the x-axis, it means the Exclusion Zone method is better.

**Figure 3.9** $f_{\text{best}}$’s difference between the Hansen and the Exclusion Zone method
Figure 3.10 CPU time difference between the Hansen and the Exclusion Zone method

Figure 3.11 Iteration difference between the Hansen and the Exclusion Zone method
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Hansen</th>
<th>Zone</th>
<th>D</th>
<th>Hansen</th>
<th>Zone</th>
<th>D</th>
<th>Hansen</th>
<th>Zone</th>
<th>D</th>
<th>Hansen</th>
<th>Zone</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>101</td>
<td>6.67E-05</td>
<td>6.67E-05</td>
<td>0</td>
<td>16</td>
<td>15</td>
<td>-1</td>
<td>19</td>
<td>12</td>
<td>-7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>102</td>
<td>1.10E-06</td>
<td>8.74E-05</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>13</td>
<td>13</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>103</td>
<td>1.32E-06</td>
<td>1.67E-05</td>
<td>0</td>
<td>31</td>
<td>31</td>
<td>16</td>
<td>19</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>104</td>
<td>2.87E-05</td>
<td>9.05E-06</td>
<td>-1.97E-05</td>
<td>15</td>
<td>0</td>
<td>-15</td>
<td>15</td>
<td>10</td>
<td>-5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>105</td>
<td>5.38E-05</td>
<td>0.0001</td>
<td>4.63E-05</td>
<td>16</td>
<td>0</td>
<td>-16</td>
<td>15</td>
<td>13</td>
<td>-2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>106</td>
<td>3.48E-05</td>
<td>4.21E-05</td>
<td>7.26E-06</td>
<td>0</td>
<td>15</td>
<td>15</td>
<td>11</td>
<td>14</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>107</td>
<td>9.40E-05</td>
<td>0.000556</td>
<td>4.62E-04</td>
<td>0</td>
<td>47</td>
<td>47</td>
<td>50</td>
<td>30</td>
<td>-20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>108</td>
<td>4.82E-05</td>
<td>4.82E-05</td>
<td>0</td>
<td>16</td>
<td>31</td>
<td>15</td>
<td>12</td>
<td>18</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>109</td>
<td>2.38E-05</td>
<td>1.81E-05</td>
<td>-5.67E-06</td>
<td>16</td>
<td>0</td>
<td>-16</td>
<td>20</td>
<td>16</td>
<td>-4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>201</td>
<td>4.57E-05</td>
<td>2.83E-05</td>
<td>-1.73E-05</td>
<td>15</td>
<td>78</td>
<td>63</td>
<td>80</td>
<td>284</td>
<td>204</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>202</td>
<td>7.85E-05</td>
<td>0.011934</td>
<td>1.19E-02</td>
<td>93</td>
<td>2964</td>
<td>2871</td>
<td>302</td>
<td>1498</td>
<td>1196</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>204</td>
<td>6.83E-05</td>
<td>7.10E-05</td>
<td>2.78E-06</td>
<td>63</td>
<td>94</td>
<td>31</td>
<td>195</td>
<td>249</td>
<td>54</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>205</td>
<td>3.82E-05</td>
<td>8.33E-05</td>
<td>4.52E-05</td>
<td>16</td>
<td>31</td>
<td>15</td>
<td>16</td>
<td>24</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>206</td>
<td>7.05E-05</td>
<td>2.45E-05</td>
<td>-4.60E-05</td>
<td>16</td>
<td>31</td>
<td>15</td>
<td>13</td>
<td>22</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>207</td>
<td>5.52E-05</td>
<td>0.01043</td>
<td>1.04E-02</td>
<td>405</td>
<td>47</td>
<td>-358</td>
<td>1328</td>
<td>40</td>
<td>-1288</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>208</td>
<td>1.94E-05</td>
<td>0.000641</td>
<td>6.21E-04</td>
<td>0</td>
<td>15</td>
<td>15</td>
<td>20</td>
<td>20</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>209</td>
<td>2.83E-07</td>
<td>1.74E-05</td>
<td>1.71E-05</td>
<td>31</td>
<td>0</td>
<td>-31</td>
<td>24</td>
<td>12</td>
<td>-12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>210</td>
<td>4.20E-05</td>
<td>3.92E-05</td>
<td>-2.81E-06</td>
<td>62</td>
<td>62</td>
<td>0</td>
<td>115</td>
<td>44</td>
<td>-71</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>211</td>
<td>3.36E-05</td>
<td>1.66E-05</td>
<td>-1.70E-05</td>
<td>32</td>
<td>203</td>
<td>171</td>
<td>295</td>
<td>567</td>
<td>272</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>212</td>
<td>8.28E-05</td>
<td>-4.35E-06</td>
<td>-8.71E-05</td>
<td>4633</td>
<td>1794</td>
<td>-2839</td>
<td>5629</td>
<td>1293</td>
<td>-4336</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>303</td>
<td>9.16E-05</td>
<td>9.16E-05</td>
<td>0</td>
<td>16</td>
<td>31</td>
<td>15</td>
<td>20</td>
<td>20</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>304</td>
<td>9.16E-05</td>
<td>8.58E-05</td>
<td>-5.79E-06</td>
<td>1341</td>
<td>45755</td>
<td>44414</td>
<td>3874</td>
<td>16327</td>
<td>12453</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>401</td>
<td>9.16E-05</td>
<td>9.16E-05</td>
<td>0</td>
<td>31</td>
<td>47</td>
<td>16</td>
<td>86</td>
<td>86</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>402</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>16</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>403</td>
<td>8.01E-05</td>
<td>8.01E-05</td>
<td>0</td>
<td>47</td>
<td>47</td>
<td>0</td>
<td>93</td>
<td>93</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>404</td>
<td>9.97E-05</td>
<td>6.95E-05</td>
<td>-3.02E-05</td>
<td>30280</td>
<td>639</td>
<td>-29641</td>
<td>22376</td>
<td>380</td>
<td>-21996</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>405</td>
<td>-9.95E-10</td>
<td>-9.95E-10</td>
<td>0</td>
<td>31</td>
<td>63</td>
<td>32</td>
<td>62</td>
<td>62</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>702</td>
<td>5.21E-06</td>
<td>5.21E-06</td>
<td>0</td>
<td>78</td>
<td>125</td>
<td>47</td>
<td>120</td>
<td>120</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.5 Detailed data results between the Hansen and the Exclusion Zone method

From the above three figures, we can conclude that by applying Exclusion Zone method, the number of iterations and the $f_{\text{best}}$ are better compared with the Standard Hansen method, although the CPU time is longer for a few cases.
When we looked at the CPU time, it is possible that Exclusion Zone method takes more time in the iteration since the machine could be overheated because of constant computation. The abnormal long CPU time will make it difficult to compare. For the number of iterations, the graph and data table show that it is better in general, and it is much better for some cases. Therefore we can conclude that the Exclusion Zone method is effective.
4.1 Introduction

Unlike non-interval algorithms, the constrained versions of interval minimization algorithms are very much similar in structure to the unconstrained version. The major differences are extra deletion conditions associated with the constrained functions, and the feasibility requirement for updating the upper bound of optimal objective function value. The extra deletion conditions could improve convergence of the Interval Arithmetic method to a certain extent. Thus the constrained version provides a positive enhancement. However, the feasibility requirement of sample points for updating the upper bound $\bar{f}$ of optimal objective function value would greatly obstruct the speed of convergence. When midpoints are not feasible, an update of $\bar{f}$ relies upon additional information. Such additional information, which can be easily obtained, is the maximum of the objective inclusion function for the remaining subintervals (called $f_{\text{best use}}$). Recently, many more sophisticated constrained strategies have been developed. Several of them are summarized in section 2.4. However, locating a feasible sample point in any compact interval domain is almost as difficult as finding an optimal solution of the original optimization problem. Thus, in this study, we only consider some special cases of constraints and develop effective procedures for locating feasible sampling points if there are any.

There are some special linear and quadratic constraints such as $\sum_{i=1}^{n} x_i = 1$, or $\sum_{i=1}^{n} x_i^2 = 1$ ($<$, $>$ as well) in several important applications of optimization. In this chapter we propose two
similar methods to accurately test the domain’s feasibility and locate a feasible point if the feasible set is nonempty in a situation of only one special equality or inequality constraint of these types.

\[ c_1 x_1 + c_2 x_2 + \ldots + c_n x_n = c_0, \]

\[ c_1 x_1 + c_2 x_2 + \ldots + c_n x_n \leq c_0, \tag{4.1.1} \]

\[ c_1 x_1^2 + c_2 x_2^2 + \ldots + c_n x_n^2 = c_0, \]

\[ c_1 x_1^2 + c_2 x_2^2 + \ldots + c_n x_n^2 \leq c_0. \]

Of course, they include special constraints with all c’s equal to 1 as in several important applications of (C-GOP), such as resource allocation. Our treatment method is also applicable to a more general separable constraint.

### 4.2 Algorithm with One Special Constraint

For one special constraint such as \( \sum_{i=1}^{n} x_i = 1, \sum_{i=1}^{n} x_i^2 = 1, \sum_{i=1}^{n} x_i \leq 1, \) or \( \sum_{i=1}^{n} x_i^2 \leq 1, \) we have developed an effective algorithm to deal with them. We use a backward iterative scheme to determine each \( x_i \) value. To illustrate the idea, let us consider \( \sum_{i=1}^{n} x_i = 1, \) for which the algorithm is outlined as follows:

1. Get the inclusion function \( H_n(X) = \sum_{i=1}^{n} X_i = [lb H_n(X), ub H_n(X)] \). Set \( s_n = 1, \) and \( k = n. \)

2. Different terminating cases:
   a. If \( lb H_k(X) > s_k, \) or \( ub H_k(X) < s_k, \) there is no feasible solution. Stop;
   b. If \( lb H_k(X) = s_k, \) or \( ub H_k(X) = s_k, \) we have found a feasible solution. Stop;
   c. If \( k = 1, \) set \( x_k = s_k \) and we have found a feasible solution. Stop.

3. Otherwise,
a. Get \( H_{k-1}(X) = \sum_{i=1}^{k-1} X_i = [lb \, H_{k-1}(X), ub \, H_{k-1}(X)] \);

b. If \( ub \, H_{k-1}(X) + lb \, X_k = s_k \), we have found a feasible solution. Stop;

c. If \( ub \, H_{k-1}(X) + lb \, X_k < s_k \), we have found a feasible solution. Stop;

d. If \( ub \, H_{k-1}(X) + lb \, X_k > s_k \), let \( x_k = lb \, X_k \), \( s_{k-1} = s_k - x_k \), and \( k = k - 1 \). Go to step 2.

**Theorem 4.1:** (finite step convergence) For any given interval \( X \) in \( \mathbb{R}^n \), the procedure above either finds a feasible point in \( X \) or determines that no feasible point exists in \( X \) after a finite number of iterations.

**Proof.** Write \( X_k = [a_k, b_k], k = 1, \ldots, n \). The possible infeasibility is detected in Step 2a. Once Step 2a is passed for \( k = n \), a feasible point is located by repeating applications of the remaining steps. The maximum number of such repeats is \( n - 1 \) according to Steps 2c and 3d. It is trivial to see that when a stop is hit at Step 2b or 3b, a feasible \( x \) is obtained. When a stop is hit at Step 2c, \( a_1 < s_1 < b_1, x = (s_1, x_2, \ldots, x_n) \) is feasible. When a stop is hit at Step 3c,

\[
ub \, H_{k-1}(X) + a_k < s_k < ub \, H_{k-1}(X) + b_k; \tag{4.1.2}
\]

Therefore, \( x = (b_1, \ldots, b_{k-1}, s_k - ub H_{k-1}(X), x_{k+1}, \ldots, x_n) \) is feasible. Finally at Step 3d, \( x_k \) is clearly feasible in the sense that the remaining variables can be determined sequentially to form a feasible point \( x \). Furthermore, we would have

\[
lb \, H_{k-1}(X) < s_{k-1} < ub \, H_{k-1}(X). \tag{4.1.3}
\]

Then the same arguments apply to this sub problem

\[
\sum_{i=1}^{k-1} x_i = s_{k-1}. \tag{4.1.4}
\]

The desired conclusion follows according to the mathematical induction. #
For $\sum_{i=1}^{n} x_i^2 = 1$, the algorithm steps can be similarly designed. The only difference is the way we need to treat $x_i^2$ instead of $x_i$ in each step of the backward iterative process. Since the domain for each $x_i, i = 1, 2, ..., n$, varies, we separate it into three cases:

1. $x_i \in (-\infty, 0), X_i^2 = [(ub X_i)^2, (lb X_i)^2]$;
2. $x_i \in (0, +\infty), X_i^2 = [(lb X_i)^2, (ub X_i)^2]$;
3. $x_i \in (-\infty, +\infty), X_i^2 = [0, \max((lb X_i)^2, (ub X_i)^2)]$.

If we add the above three cases into the algorithm, we can obtain a similar algorithm for $\sum_{i=1}^{n} x_i^2 = 1$. In particular, the above algorithm could be extended to $\sum_{i=1}^{n} c_i x_i = c_0$ and $\sum_{i=1}^{n} c_i x_i^2 = c_0$ with some minor modifications. This procedure can be extended further to other types of inequality constraint involving a single separable constraint function. The inequality $\sum_{i=1}^{n} x_i \leq 1$ can be treated in a similar methodology like the following algorithm.

1. Get the inclusion function $G_n(X) = \sum_{i=1}^{n} X_i = [lb H_n(X), ub H_n(X)]$. Set $s_n = 1$, and $k = n$.

2. Different terminating cases:
   a. If $lb G_k(X) > s_k$, there is no feasible solution. Stop; If $ub G_k(X) < s_k$, choose the mid ($x_i$), we have found a feasible solution. Stop;
   b. If $lb G_k(X) = s_k$, or $ub G_k(X) = s_k$, we have found a feasible solution. Stop;
   c. If $k = 1$, set $x_k = s_k$ and we have found a feasible solution. Stop.

3. Otherwise,
   a. Get $G_{k-1}(X) = \sum_{i=1}^{k-1} X_i = [lb G_{k-1}(X), ub G_{k-1}(X)]$;
   b. If $ub G_{k-1}(X) + lb X_k = s_k$, we have found a feasible solution. Stop;
   c. If $ub G_{k-1}(X) + lb X_k < s_k$, we have found a feasible solution. Stop;
d. If \( ub \ G_{k-1}(X) + lb \ X_k > s_k \), let \( x_k = lb \ X_k \), \( s_{k-1} = s_k - x_k \), and \( k = k - 1 \). Go to step 2.

Therefore, the difference between the equation constraint’s algorithm and inequality constraint’s algorithm is step 2a. For the inequality constraint, we can’t delete if \( ub \ \ G_k(X) < s_k \) and all the points are feasible in the given domain. Here is a simple example to illustrate the process.

**Example 4.1:**

\[
\min f(x_1, x_2) = [x_1^2 - 0.3\cos(3\pi x_1)] + [2x_2^2 - 0.4\cos(4\pi x_2)] + 0.7
\]

subject to \( 2x_1 - x_2 \leq 0 \),

\( x_i \in [-1,1.5] \), for \( i = 1, 2 \).

From the above optimization problem, we could easily find out that the global feasible solution satisfying the constraint would be \( x^* = (0,0) \) with \( f^* = 0 \). After applying the procedure above, we got the numerical result as follows:

\[
f_{best} = 3.87422 \times 10^{-5};
\]

\[
x_{best} = [-0.00146484, -0.000488281].
\]

Taking this \( x \) back to the constraint to double check, we have

\[
2x_1 - x_2 = -1.00195 \leq 0.
\]

Thus the given inequality is satisfied. #

With more numerical experiments, we conclude that this procedure has a tendency to locate a feasible point among the vertices of the interval. This tendency may not be desirable for interval methods since many subintervals generated by Interval Arithmetic method may share the
same feasible vertices, which leads to duplication and waste. Our numerical test results also confirmed this possible scenario. Thus we modified the procedure as described below.

We decided to use the midpoints as much as possible. Suppose the interval for each variable $x_i$ is $[a_i, b_i]$. Let $m_i$ represent each variable’s midpoint. We use the following steps to depict the algorithm.

1. If $\sum m_i = s_k$, a feasible point is found and stop;
2. If $\sum m_i < s_k$, look at the interval $\prod [m_i, b_i]$. Let $m_i^+$ be the midpoint of this new interval.
   a. If $\sum m_i^+ = s_k$, a feasible point is found and stop;
   b. If $\sum m_i^+ < s_k$, solve over the interval $\prod [m_i^+, b_i]$ by using the above procedure; If $\sum b_i > s_k$, let $\delta = \frac{\sum b_i - s_k}{n}$. Write $\delta_i = \min(\delta, b_i - m_i^+)$. Solve over the interval $\prod [m_i^+, b_i - \delta_i]$ by using the above procedure.
   c. If $\sum m_i^+ > s_k$, solve over the interval $\prod [m_i, m_i^+]$ by using the above procedure.
3. If $\sum m_i > s_k$, look at the interval $\prod [a_i, m_i]$. Let $m_i^-$ be the midpoint of this new interval.
   a. If $\sum m_i^- = s_k$, a feasible point is found and stop;
   b. If $\sum m_i^- < s_k$, solve over the interval $\prod [m_i^-, m_i]$ by using the above procedure;
   c. If $\sum m_i^- > s_k$, solve over the interval $\prod [a_i, m_i^-]$ by using the above procedure; If $\sum a_i < s_k$, let $\delta = \frac{s_k - \sum a_i}{n}$. Write $\delta_i = \min(\delta, m_i^- - a_i)$. Solve over the interval $\prod [a_i + \delta_i, m_i^-]$ by using the above procedure.

A class of examples which are applied by above algorithm including improvements are shown in the following section.
4.3 A Class of Examples with One Special Constraint

As an application of the procedures presented in section 4.1, we consider one class of examples that is defined as

\[
\min f(x) = x^T Ax
\]
subject to \( \sum_{i=1}^{n} x_i^2 = 1 \) \quad (4.3.1)
\[
x_i \in [-1, 1], \text{ for } i = 1, \ldots, n.
\]

where \( A \) is any symmetric matrix. This constrained minimization problem can be interpreted as the minimum eigenvalue problem.

**Proposition 4.1:** let \( \lambda^* \) be the smallest eigenvalue of the given symmetric matrix \( A \), and \( f^* \) the smallest objective function value. Then \( \lambda^* = f^* \).

**Proof:** Let \( \lambda_1, \lambda_2, \ldots, \lambda_n \) be all the real eigenvalues of the matrix \( A \), with \( \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n \). For each \( i \), let \( e_i \) be a normalized eigenvector associated with \( \lambda_i \).

Since

\[
A e_i = \lambda_i e_i, e_i^T e_i = 1, \quad e_i \in [-1, 1], \text{ for } i = 1, 2, \ldots, n,
\]

we have

\[
e_i^T A e_i = \lambda_i e_i^T e_i = \lambda_i.
\]

Thus we could conclude that

\[
f^* \leq e_i^T A e_i = \lambda_i, \text{ for } i = 1, 2, \ldots, n,
\]

from which we could get

\[
f^* \leq \lambda_1.
\]

On the other hand, we have \( \forall x \), there is

\[
x^T A x \geq \lambda_1 x^T x.
\]

This implies for any feasible \( x \)
Thus

\[ f^* \geq \lambda_1. \]  

(4.3.8)

Therefore

\[ f^* = \lambda_1 = \lambda^*. \]  

(4.3.9)

### 4.4 Numerical Results: One Special Constraint Case

In this section, we start with an example with comparison in original algorithm and in modified version, then describe the result of applying the algorithm in modified version to several multi-dimensional problems, in each of which the actual minimum objective function value should be explicitly known by simple matrix algebra. For convenience purpose, we have already shifted the best objective function value to zero in the following examples.

**Example 4.2:** Let \( A = \begin{bmatrix} 5 & 4 \\ 4 & 5 \end{bmatrix} \), then this question is

\[
\begin{align*}
\min f(x) &= 5x_1^2 + 8x_1x_2 + 5x_2^2 - 1 \\
\text{subject to} & \quad \sum_{i=1}^{2} x_i^2 = 1 \\
& \quad x_i \in [-1,1], \text{ for } i = 1, 2.
\end{align*}
\]

Here is the table showing the effectiveness of the modified version.

<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_{best} )</td>
<td>( 9.12696 \times 10^{-8} )</td>
<td>( 2.98023 \times 10^{-8} )</td>
</tr>
<tr>
<td>CPU(ms)</td>
<td>124</td>
<td>62</td>
</tr>
<tr>
<td>Iteration</td>
<td>330</td>
<td>166</td>
</tr>
<tr>
<td>( x_{best} )</td>
<td>((-0.707031, 0.707182))</td>
<td>((-0.70764, 0.70715))</td>
</tr>
</tbody>
</table>

Table 4.1 Data comparison between the original algorithm and the modified algorithm

Similarly, more examples are demonstrated below, where we describe:

a) the original function \( f(x) \), including the matrix form and the extended form;
b) the initial variables’ domain;

c) the actual minimum eigenvalue $\lambda^*$ and the actual minimum $f^*$;

d) the value $f_{best}$ and $x_{best}$ produced by the algorithm;

e) the number of iterations and the CPU time in seconds;

f) the stopping ID which shows how the results reach the optimal answer, and the stopping
error in all the following examples is $10^{-5}$.

Examples 4.3--4.6:

4.3 $\min f(x) = x^T \begin{bmatrix} 3 & 1 \\ 1 & 3 \end{bmatrix} x - 2$

\[= 3x_1^2 + 2x_1x_2 + 3x_2^2 - 2\]

- $\sum_{i=1}^{2} x_i^2 = 1, x_i \in [-1,1]$, for $i = 1, 2$.
- $\lambda^* = 2, f^* = 0$.
- $f_{best} = 4.76849 \times 10^{-5}$, $x_{best} = (0.707, -0.707)$.
- 124 iterations, CPU time = 78 ms.
- stop id: due to $f_{best} = 4.76849 \times 10^{-5} \leq f^* = 0$ in absolute error tolerance 0.0001.

4.4 $\min f(x) = x^T \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} x - 1$

\[= 2x_1^2 + 2x_2^2 + 2x_3^2 + 2x_1x_2 + 2x_1x_3 + 2x_2x_3 - 1\]

- $\sum_{i=1}^{3} x_i^2 = 1, x_i \in [-1,1]$, for $i = 1, 2, 3$.
- $\lambda^* = 1, f^* = 0$.
- $f_{best} = 4.12952 \times 10^{-6}$, $x_{best} = (-0.816, 0.577, -0.324)$.
- 351 iterations, CPU time = 218 ms.
- stop id: due to $f_{best} = 4.12952 \times 10^{-6} \leq f^* = 0$ in absolute error tolerance 0.0001.
4.5 min \( f(x) = x^T \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 2 \\ 3 & 2 & 1 \end{bmatrix} x + 2 \)

\[ \Rightarrow x_1^2 + x_2^2 + x_3^2 + 4x_1x_2 + 6x_1x_3 + 4x_2x_3 + 2 \]

- \( \sum_{i=1}^{3} x_i^2 = 1, x_i \in [-1,1], \text{for } i = 1, 2, 3. \)
- \( \lambda^* = -2, f^* = 0. \)
- \( f_{best} = 8.92034 \times 10^{-5}, x_{best} = (-0.606, 0.707, 0.365). \)
- 9987 iterations, CPU time = 15s 366ms.
- stop id: due to \( f_{best} = 8.92034 \times 10^{-5} \leq f^* = 0 \) in absolute error tolerance 0.0001.

4.6 min \( f(x) = x^T \begin{bmatrix} 1 & -2 & 1 & -1 \\ -2 & 1 & -2 & 1 \\ 1 & -2 & 1 & -2 \\ -1 & 1 & -2 & 1 \end{bmatrix} x + 1.618 \)

\[ \Rightarrow x_1^2 + x_2^2 + x_3^2 + x_4^2 - 4x_1x_2 + 2x_1x_3 - 2x_1x_4 - 4x_2x_3 + 2x_2x_4 - 4x_3x_4 + 1.618 \]

- \( \sum_{i=1}^{4} x_i^2 = 1, x_i \in [-1,1], \text{for } i = 1, 2, 3, 4. \)
- \( \lambda^* = -1.618, f^* = 0. \)
- \( f_{best} = 0.000343303, x_{best} = (0.540, 0.372, 0.602, 0.457). \)
- 52399 iterations, CPU time = 30m 6s 725ms.
- stop id: iteration limits.

From above examples, we can see the effectiveness of the algorithm by the following parameters:

1. \( f_{best} \). All the \( f_{best} \) we got from above 4 examples are close enough to the corresponding minimum eigenvalue.

2. Number of iterations. The iteration number in examples with a dominant diagonal matrix is much less than the iteration number in examples with a non-dominant diagonal matrix. This
means the algorithm works well for dominant diagonal matrix problem. For non-dominant diagonal matrix, it ultimately will find a good solution with more time and iterations.

3. CPU Time. The cost CPU time for each example is under the maximum limit. Especially for the first three examples, it implies that the algorithm is effective.
CHAPTER 5

PROCEDURES DEALING WITH TWO LINEAR CONSTRAINTS

This chapter deals with the two linear constraints case. Presence of multiple constraints creates new obstacles and challenges in the feasibility sampling. For example, the finite step convergence of the sequential algorithm presented in the previous chapter is no longer available. In fact, the algorithm has to be completely redesigned. Three algorithms are presented in this chapter.

5.1 Coordinate Decent method

The coordinate decent method is easy to implement. However, its convergence property is poor. At any given point \( x = (x_1, x_2, ..., x_n) \), the method solves

\[
\min_{x_i} f (x_1, x_2, x_3 \ldots x_n).
\]

(5.1.1)

with respect to single coordinate variable \( x_i \) chosen in a listed method: the \( x_1 \) is changed first, then \( x_2 \) and so forth through \( x_n \). The process is then repeated starting with \( x_1 \) again if the objective function is not small enough.

In this method, we choose the midpoint as the starting sampling point. We introduced two objective functions for our feasibility sampling problem. Let \( v_1 \) and \( v_2 \) be the difference between the desired value of each constraint and its estimation value. Then define the following two versions of the objective function for our feasibility sampling problem

\[
V_{\text{max}} (x) = \max \{|v_1|, |v_2|\};
\]

(5.1.2)
\( V_{\text{sum}}(x) = |v_1| + |v_2|. \)  

Thus a point \( x \) is feasible if and only if both \( v_1 \) and \( v_2 \) are zero. We separate the results into several different cases. Based on the difference between the desired value of the constraints and the estimation values \( v_1, v_2 \), we could generate many cases,

1. Both \( v_1 \) and \( v_2 \) are greater than 0.
2. Both \( v_1 \) and \( v_2 \) are less than 0.
3. One is greater than 0, the other one is less than 0.
4. One is equal to 0, the other one is not.
5. Both \( v_1 \) and \( v_2 \) are equal to 0; this means we have found the answer.

Above situations depend on variables’ coefficient as well. Our goal is trying to make both violations \( v_1 \) and \( v_2 \) smaller at the same time (referred to as “no confliction” below) if possible. In order to do so, we consider as the following different situations;

1. No confliction. Both violations \( v_1 \) and \( v_2 \) can be improved at the same time.
2. Confliction. Violations \( v_1 \) and \( v_2 \) can’t be improved at the same time.

We use the following graphs to demonstrate the different situations. In each case of these graphs, the horizontal line represents the zero violation, the two circles represent the violations for the two constraints (first constraint on the left), if a circle is above the line, this means the violation is greater than 0; if the circle is blow the line, the violation is less than 0; if the circle is on the line, the violation is 0. The symbol +, or - inside the circles represent the sign of \( x_i \)’s coefficients \( c_{1i}, c_{2i} \) in the two constraints. If a circle has double symbols inside, this means this coefficient dominates compared with the other one. The dash line with an arrow shows that the corresponding variable \( x_i \) needs to go up or down in order to improve the feasibility. The \( \Delta x_i \) represents the magnitude of change for each \( x_i \). If the dash line goes up, the generated \( \Delta x_i \) is the
actual change. If the dash line goes down, then the generated $\Delta x_i$ multiplied by the negative sign is the actual change.

1. No conflict: $v_1 v_2 > 0$ (case A, B, C, D). $v_1 v_2 < 0$ (case E, F, G, H). $\Delta x_i = \min(|v_1/c_{1i}|, |v_2/c_{2i}|)$. Both $V_{\text{max}}$ and $V_{\text{sum}}$ would decrease.

![Figure 5.1 Coordinate descent in no conflict cases](image)

2. Conflict. For the conflict situation, we have several different types of cases:

Type 1: $v_1 v_2 = 0$. See figure 5.2.

a. The coefficient for the constraint with nonzero violation dominates. So $|c_{1i}| \leq |c_{2i}|$ for cases I, J, K, L, $\Delta x_i = |v_2/c_{2i}|$; $|c_{1i}| \geq |c_{2i}|$ for cases M, N, O, P, $\Delta x_i = |v_1/c_{1i}|$. Both $V_{\text{max}}$ and $V_{\text{sum}}$ would decrease.
b. The coefficient for the constraint with zero violation dominates. So \( |c_{1i}| \geq |c_{2i}| \) for cases I, J, K, L, \( \Delta x_i = \frac{|v_2|}{|c_{2i}|} \); \( |c_{1i}| < |c_{2i}| \) for cases M, N, O, P, \( \Delta x_i = \frac{|v_1|}{|c_{1i}|} \). So \( V_{max} \) decreases, but \( V_{sum} \) would increase.

![Coordinate descent in conflict cases of type 1](image)

**Figure 5.2 Coordinate descent in conflict cases of type 1**

Remark: For the above cases, they can be applied in cases with different distances away from horizontal line. But for the following cases, the distances away from the horizontal line are addressed. Distances away from horizontal line represent the magnitude of violations.

**Type 2:**
a. $v_1 v_2 < 0$ with equal distances, and both coefficients are of the same sign. For case Q, S,

$\Delta x_i = |v_i|_{c_i l}$, else is $\Delta x_i = |v_2|_{c_i l}$. So $V_{\text{max}}$ increases, but $V_{\text{sum}}$ decreases.

![Figure 5.3 Coordinate descent in conflict cases of type 2a](image)

b. $v_1 v_2 > 0$ with equal distances. Coefficients are of different signs. For case AA, BB, CC,

$\Delta x_i = |v_1|_{c_i l}$, else is $\Delta x_i = |v_2|_{c_i l}$. So $V_{\text{max}}$ increases, but $V_{\text{sum}}$ decreases.
c. \( v_1 v_2 > 0 \), the violation with dominant coefficient is closer to zero. Coefficients are of different signs. For case II, KK, MM, DD, \( \Delta x_i = \left| \frac{v_2}{c_{2i}} \right| \), else is \( \Delta x_i = \left| \frac{v_1}{c_{1i}} \right| \). So \( V_{\text{max}} \) decreases, but \( V_{\text{sum}} \) increases.
Figure 5.5 Coordinate descent in conflict cases of type 2c

d. \( v_1 v_2 > 0 \), the violation with non-dominant coefficient is closer to zero. Coefficients are of different signs. For case QQ, SS, UU, WW, \( \Delta x_i = |\frac{v_1}{c_{1i}}| \), else is \( \Delta x_i = |\frac{v_2}{c_{2i}}| \). So \( V_{\text{max}} \) decreases, and \( V_{\text{sum}} \) decreases.
e. $v_1v_2 < 0$, coefficients are the same. The distances may or may not be equal. See the following graph. For case YY, ZZ, 1, 2, $\Delta x_i = \max[|v_1|, |v_2|]/|c_{2i}|$. So $V_{max}$ decreases, but $V_{sum}$ may decrease, or may increase. For case 3, 4, 5, or 6, no feasible solution could be found, and delete the interval.
Figure 5.7 Coordinate descent in conflict cases of type 2e

From above cases, we noticed that we couldn’t always make both $V_{max}$ and $V_{sum}$ decrease at the same time. From the above cases, we observe increasing $V_{sum}$ occurs less frequently. So we choose $V_{sum}$ as our objective value in the process. Since we can’t guarantee $V_{sum}$ will decrease in cases 1b, 2c and 2e, we can apply $\Delta x_i = (|v_1| + |v_2|)/(|c_{1i}| + |c_{2i}|)$ to make the new violations $v_1$ and $v_2$ have an equal magnitude. By this approach, we can obtain cases 1a or 2a or 2b. Then apply the adjustment for each case listed in the above figures at this stage, we can make sure the $V_{sum}$ decreases (it is shown on the figures above). Since we choose the midpoint as our sampling point in coordinate descent method, it may not be able to go through the feasible point, so it is not an ideal point. In this case it may take a long time to find out a feasible point, which is a drawback for this method.

The overall feasibility sampling algorithm can be stated as follow:

Initialization: use the midpoint as the initial sampling point.
Update: for $i = 1, 2, ..., n$

Step 1: get two constraints’ variations $v_1$ and $v_2$;

Step 2: determine whether there is a conflict, or not. Then define which case;

Step 3: get the corresponding adjustment $\Delta x_i$;

Step 4: set $x_i = x_i + \Delta x_i$;

Step 5: check that if $x_i$ is still inside the domain. If it is beyond the upper bound,

$$x_i = ub X_i;$$

if it is less than the lower bound, $x_i = lb X_i$.

Stopping condition: if the current sample is not good enough, go to the update step.

The global convergence for the coordinate descent method in the dissertation is not fully verified. There are a number of references addressing the convergence issue of the coordinate descent method under different circumstances. For example, in [50], a weak convergence of the coordinate descent method for minimizing differentiable functions has been proved with an additional assumption that a unique solution exists for the minimization in each direction at any stage of the algorithm. There is another paper [53] showing convergence of the coordinate descent method with different assumptions (mainly the separable non-smooth component).

However, both existing convergence results (as well as many others) do not apply to our situation because their assumptions are not totally satisfied for our examples. So the convergence for this method could be potentially very poor (see section 5.4 for test results). From the data comparison shown in section 5.4, we conclude that our coordinate descent method works in lower dimensional cases. The convergence decays as the problem dimension increases.

5.2 Cutting Line method

The approach of addressing two linear constraints of a problem simultaneously is inspired
by Affine Arithmetic and its applications. As stated in [3], a quantity \( x \) could be represented by an expression of the form

\[
\hat{x} = x_0 + x_1 \varepsilon_1 + x_2 \varepsilon_2 + \cdots + x_n \varepsilon_n,
\]

(5.2.1)

where \( \varepsilon_i \) is the noise symbol and for each \( \varepsilon_i \) is within the interval \([-1, 1]\). We made a small amendment such that a partially unknown quantity \( x \) is represented by form \( \bar{x} \), which is a first-degree polynomial

\[
\bar{x} = x_0 + \varepsilon_1 x_1 + \cdots + \varepsilon_n x_n,
\]

(5.2.2)

where the \( \varepsilon_i \)'s are known real coefficients, and the \( x_i \)'s are symbolic variables whose value are unknown but assumed to lie in the interval \([-1,1]\).

Similarly, when it comes to two linear constraints like following

\[
a_0 + a_1 x_1 + \cdots + a_n x_n = a; \]

(5.2.3)

\[
b_0 + b_1 x_1 + \cdots + b_n x_n = b,
\]

(5.2.4)

we can apply the same methodology. First of all, we need to normalize all \( x_i \)'s domains into \([-1, 1]\). Suppose we are given an ordinary interval for \( x_i \)'s domain, which is \([lb X_i, ub X_i]\). We can generate from it a valid form for the same quantity with the following formula:

\[
\bar{x}_i = \frac{2}{(ub x_i - lb x_i)}(x_i - \text{mid}(X_i)).
\]

(5.2.5)

Introducing two unknown quantities \( \alpha, \) and \( \beta, \) we can obtain our new linear constraints:

\[
a'_0 + a'_1 \bar{x}_1 + \cdots + a'_n \bar{x}_n = \alpha; \]

(5.2.6)

\[
b'_0 + b'_1 \bar{x}_1 + \cdots + b'_n \bar{x}_n = \beta,
\]

(5.2.7)

where \( a'_i, b'_i \) are new real coefficients generated from (5.2.5) and \( \bar{x}_i = [-1,1] \) for \( i = 1,2,3,\ldots,n \).

The \( \bar{x}_i \) symbolizes the uncertainty in the value of \( \alpha, \beta, \) the magnitude and sign of the dependency.
is determined by the corresponding coefficients $a'_i, b'_i$. Thus we have successfully changed the multi-dimensional problem into two-dimensional problem in $\alpha - \beta$ coordinate system.

**Example 5.1.** We are given two linear constraints as follows:

\begin{align*}
    x_1 + x_2 + x_3 + x_4 &= 1; \quad (5.2.7) \\
    x_1 + 2x_3 + 3x_4 &= 3. \quad (5.2.8)
\end{align*}

$x_i$'s' domains are $[-2, 0] \times [-2, 2] \times [-2, 2] \times [-2, 2]$. After normalizing the domains, we get the following new system:

\begin{align*}
    \bar{x}_1 + 2\bar{x}_2 + 2\bar{x}_3 + 2\bar{x}_4 - 1 &= 1; \quad (5.2.9) \\
    \bar{x}_1 + 4\bar{x}_3 + 6\bar{x}_4 - 1 &= 3. \quad (5.2.10)
\end{align*}

Suppose two quantities $\alpha, \beta$ are represented by the forms

\begin{align*}
    \alpha &= \bar{x}_1 + 2\bar{x}_2 + 2\bar{x}_3 + 2\bar{x}_4 - 1; \quad (5.2.11) \\
    \beta &= \bar{x}_1 + 4\bar{x}_3 + 6\bar{x}_4 - 1. \quad (5.2.12)
\end{align*}

From this we can tell that the feasible value $\alpha$ lies in the interval $[-8, 6]$, and $\beta$ lies in the interval $[-12, 10]$. However, since they both include the same variables $\bar{x}_1$, $\bar{x}_3$, and $\bar{x}_4$ with nonzero coefficients, they are not entirely independent of each other. In fact, the pair $(\alpha, \beta)$ is constrained to lie in the region of $\mathbb{R}^2$ depicted in figure 5.8, which is much reduced than the rectangle $[-8,6] \times [-12,10]$ obtained through the standard interval arithmetic.

Next, we use the elimination method to cancel each variable to get an equation in terms of $\alpha, \beta$. For $\bar{x}_1$, use equation (5.2.12) minus equation (5.2.11) to get

\[ \beta = \alpha + 0 - 2\bar{x}_2 + 2\bar{x}_3 + 4\bar{x}_4. \quad (5.2.13) \]

By using the interval properties, $-2\bar{x}_2 + 2\bar{x}_3 + 4\bar{x}_4 \in [-8,8]$, thus, we get two parallel lines $\beta^-_1 = \alpha - 8, \beta^+_1 = \alpha + 8$, which are $\bar{x}_1$'s boundary lines. For $\bar{x}_2$, it's different from $\bar{x}_1$, because
it’s missing in equation (5.2.12). Under this circumstance, we only look at equation (5.2.12) and rewrite it as

\[ \beta = 0\alpha - 1 + \bar{x}_1 + 4\bar{x}_3 + 6\bar{x}_4. \]  \hspace{1cm} (5.2.14)

Again, \( \bar{x}_1 + 4\bar{x}_3 + 6\bar{x}_4 \in [-11,11] \). We get another two lines: \( \beta^+_2 = -12, \) and \( \beta^+_2 = 10 \). These are boundary lines for \( \bar{x}_2 \). Similarly, we could also get these two boundary lines for \( \bar{x}_3 \):

\[ \beta = 2\alpha + 1 - \bar{x}_1 - 4\bar{x}_2 + 2\bar{x}_4. \]  \hspace{1cm} (5.2.15)

\[ \beta^+_3 = 2\alpha - 6, \beta^-_3 = 2\alpha + 8. \]  \hspace{1cm} (5.2.16)

For \( \bar{x}_4 \):

\[ \beta = 3\alpha + 2 - 2\bar{x}_1 - 6\bar{x}_2 - 2\bar{x}_3. \]  \hspace{1cm} (5.2.17)

\[ \beta^+_4 = 3\alpha - 8, \beta^-_4 = 3\alpha + 12. \]  \hspace{1cm} (5.2.18)

After we plug all these 8 lines in the \( \alpha - \beta \) coordinate system, we could obtain a boundary region shown in figure 5.8.
Figure 5.8 Boundary region

The above steps are called bounding steps. Next step is to get the feasible value of variable \( x \) for any given image point within the boundary region. We already normalized the variables’ domains into \([-1, 1]\). Then we could use one of eight boundary lines to cut through the given point \((a, b)\). Suppose we use \( x_j \)' boundary line to approach point \((a, b)\), then for the rest \( x_i, i \neq j \) in \( x_j \)' boundary line remainder, we could change only one \( x_i \)'s value to get the given point \((a, b)\). Suppose there exists \(-1 \leq t_i \leq 1\) that cuts interval \([-1, 1]\) into two subintervals: the lower box \([-1, t]\) and the upper box \([t, 1]\), and at this \( t_i \), one of \( \bar{x}_i \)'s boundary line can go through the given point \((a, b)\). If we deal with these two subintervals individually by repeating the above projection steps in order to build a new boundary region, we could obtain the following result.
Proposition 5.1: The old $x_i$’s lines in $\alpha - \beta$ coordinate system under the original box are the same as the new $x_i$’s lines in $\alpha - \beta$ coordinate system under the lower box and the upper box when $x_i$’s interval $[-1, 1]$ is cut.

Proof: Let’s explain in two different parts:

Part 1: $x_i$’s lower child box $[-1, t]$ after normalization becomes $[-1, 1]$, and the new $x_i$ is

$$x_i' = -1 + 2(x_i + 1)/(t + 1).$$

(5.2.19)

Thus

$$x_i = -1 + (t + 1)(x_i' + 1)/2.$$  

(5.2.20)

The old $x_i$’s lines in $\alpha - \beta$ coordinate system under the original box:

$$\alpha = a_0 + a_1x_1 + [a_2x_2 + \cdots + a_nx_n];$$  

(5.2.21)

$$\beta = b_0 + b_1x_1 + [b_2x_2 + \cdots + b_nx_n].$$  

(5.2.22)

By eliminating $x_i$, writing $ba_j = b_j/a_j$, $ab_j = a_j/b_j$, and assuming $a_j \neq 0$, we obtain

$$\beta = [b_0 - a_0ba_j] + ba_j\alpha + \{[b_1 - a_1ba_j]x_1 + \cdots + [b_1 - a_1ba_j]x_n\}. $$

(5.2.23)

Note: If $a_j = 0$, but $b_j \neq 0$, we need to consider $\alpha$—intercept, which is

$$\alpha = [a_0 - b_0ab_j] + ab_j\beta + \{[a_1 - b_1ab_j]x_1 + \cdots + [a_n - b_nab_j]x_n\}. $$

(5.2.24)

The new $x_i$’s lines in $\alpha - \beta$ coordinate system under the lower child box:

$$\alpha = a_0 + a_1[-1 + (t + 1)(x_i' + 1)/2] + [a_2x_2 + \cdots + a_nx_n];$$  

(5.2.25)

$$\beta = b_0 + b_1[-1 + (t + 1)(x_i' + 1)/2] + [b_2x_2 + \cdots + b_nx_n].$$

(5.2.26)

Simplify the above system, we could get

$$\alpha = a_0 - a_1 + \frac{a_1(t+1)}{2}x_i' + \frac{a_1(t+1)}{2}x_i + [a_2x_2 + \cdots + a_nx_n];$$

(5.2.27)

$$\beta = b_0 - b_1 + \frac{b_1(t+1)}{2}x_i' + \frac{b_1(t+1)}{2}x_i + [b_2x_2 + \cdots + b_nx_n].$$

(5.2.28)

which lead to
\[
\alpha = \left[ a_0 + \frac{a_1(t-1)}{2} + \frac{a_1(t+1)x'_1}{2} + [a_2x_2 + \cdots + a_nx_n] \right]; \quad (5.2.29)
\]
\[
\beta = \left[ b_0 + \frac{b_1(t-1)}{2} + \frac{b_1(t+1)x'_1}{2} + [b_2x_2 + \cdots + b_nx_n] \right]. \quad (5.2.30)
\]

Similarly by eliminating \(x_i\), we could get
\[
\beta = \left[ b_0 - a_0b_j \right] + b_j\alpha + \left\{ [b_1 - a_1b_j]x_1 + \cdots + [b_1 - a_1b_j]x_n \right\}. \quad (5.2.31)
\]

Note: If \(a_j = 0\), but \(b_j \neq 0\), we need to consider \(\alpha\) — intercept, which is
\[
\alpha = \left[ a_0 - b_0ab_j \right] + ab_j\beta + \left\{ [a_1 - b_1ab_j]x_1 + \cdots + [a_1 - b_1ab_j]x_n \right\}. \quad (5.2.32)
\]

Therefore we could get conclusion that the old \(x_i\)’s lines in \(\alpha - \beta\) coordinate system under the original box are the same as the new \(x_i\)’s lines in \(\alpha - \beta\) coordinate system under the lower box.

For the part 2 dealing with the upper box \([t, 1]\), we could repeat the same steps above to reach the same desired conclusion.

Therefore, we can conclude that if we could find a \(t\) for one variable which one line can cut through the given point \((a, b)\), then we could obtain the other variables’ value easily. However it is possible that we cannot get one variable’s \(t\) for a boundary line to cut through point \((a, b)\). In this case, we need to change to another variable to cut until it all variables are checked. If none of them makes a perfect cut. We can accept the best possible cut and repeat the process.

From proposition 5.1, we can conclude that the new enclosed region is smaller than the old feasible region. By repeating these steps, we can make sure the feasible region gets smaller and smaller. However, the finite step convergence is not verified yet since the changes in each step are not the same. So this is our future work to verify the convergence.

**Example 5.2:** (example 5.1 continued, applied in cutting line method)

First of all, get all the distances from the boundary lines to the given point \((1, 3)\),
\[
\bar{x}_1^- = 7.071, \quad \bar{x}_1^+ = 4.243;
\]
where - means lower boundary line, and + means upper boundary line. By choosing the smallest distance which is \( \bar{x}_4^- = 2.53, \bar{x}_4^+ = 3.795 \),

\[
\bar{x}_2^- = 15, \bar{x}_2^+ = 7; \quad (5.2.32)
\]
\[
\bar{x}_3^- = 3.13, \bar{x}_3^+ = 3.13; \quad (5.2.32)
\]
\[
\bar{x}_4^- = 2.53, \bar{x}_4^+ = 3.795.
\]

Thus a complete cutting line algorithm can be stated as follows:

Step 1: for \( i = 1, 2 \ldots n \), normalize variable \( x_i \)'s domain into \([-1, 1]\), get all new variables \( \bar{x}_i \) in terms of \( x_i \). Initialize ordered pair \( (a, b) \).

Step 2: generate the new system of two equations in terms of \( \{ \bar{x}_i \} \), substitute the right hand side of the new equations’ value by \( \alpha, \beta \).

Step 3: for \( i = 1, 2 \ldots n \), use the elimination method to cancel variable \( \bar{x}_i \), and the resulting equation would determine two bounded parallel lines.
Step 4: for \( i=1, 2...n \), check the point \((a, b)\) whether stays in the boundary region. If not, delete the interval.

Step 5: for \( i=1, 2...n \), get a reasonable value \( t \), two different situations:

a). \( t_i \in [-1,1] \), we can find all other \( \bar{x}_i \).

b). \( t_i \notin [-1,1] \), determine whether \( \bar{x}_i = -1 \) or \( 1 \). Then rebuild the system of equations of equations of the remaining variables, and repeat step 4 by re-ordering the variables’ sequence except \( \bar{x}_i \) until we get a \( t_j \in [-1,1], j \neq i \). If two variables left in the system of equations, two different situations:

I). the coefficients and constant are proportional. Call the modified algorithm we developed for one constraint in Chapter 4.

II). otherwise, solve it directly.

Step 6: for \( i=1, 2...n \), un-normalize \( \bar{x}_i \)’s value back to the its own real domain.

5.3 Projection Coordinate Descent method

This method is applied after bounding steps to enclose the feasible region by using the cutting line method. Since we already bounded the image of \( h(x) \) in \( \alpha - \beta \) coordinate system by two different parallel lines corresponding to every variable \( x_i \), and the feasible point \((a, b)\) is given, if we know the distance between the point \((a, b)\) and the two parallel lines, we can use the projection to estimate the cut location \( t \) in \([-1, 1]\) for that variable. One disadvantage about this projection step is that the feasibility is not ensured. However, the algorithm at least leads to a point that is very close to the feasible region in the original x-coordinate system. Once we get this point, we can reapply coordinate descent method to get closer and closer to the given point.
(a, b). Thus after using cutting line method to get parallel lines, we apply the projection coordinate descent method to get a better starting point than the midpoint used in coordinate descent method. The process is followed by coordinate descent method to obtain the feasible sampling point. This overall method is designed to accelerate convergence speed. This is expected to be faster than the coordinate descent method alone.

We can plot point (a, b) in the $\alpha - \beta$ coordinate system. If we still use the example 5.1 in section 5.2, this point would be (1, 3). Then, for each $x_i$ we could graph a line going through this point that is parallel to the $\tilde{x}_i$ boundary lines. Then we can get two intersection points of the new interior parallel line and boundary lines in the feasible region. Since all $\tilde{x}_i$’s domains are $[-1,1]$, and it is not difficult to get the distance from point (a, b) to every boundary line, we can apply the projection (shown in example 5.3 below) to get the $\tilde{x}_i$ value. From our example, we only choose $\tilde{x}_1$ as the variable to get through point (1, 3). Other variables can be simulated at the same way. See figure 5.9.
Figure 5.9 Feasible line going through the boundary region

The point we find in this approach may be infeasible. But it is close enough to the feasible region compared with the midpoint we used as the starting point in the coordinate descent method presented in section 5.1. So we could again apply the coordinate descent method to further eliminate the constraint variation.

**Example 5.3**: (example 5.1 continued)

If we plot an interior line goes through (1, 3), we can find that it intersects with two boundary lines. For example, plot the $\bar{x}_1$ line goes through (1, 3) as stated in figure 5.9. This line intersects at $\bar{x}_4$’s lower boundary line and $\bar{x}_4$’s upper boundary line with distances

$$\bar{x}_4^- = 2.53, \bar{x}_4^+ = 3.795,$$

while the distance between two parallel boundary lines is

$$d = 6.32.$$
Thus we get the portion for $\bar{x}_1$ line is
\[
p = \frac{2.53}{6.32}.
\]
For these two intersection points, it is guaranteed that on one line $\bar{x}_1 = -1$, and on the other one $\bar{x}_1 = 1$. Thus for variable $\bar{x}_1$, the distance between two parallel boundary lines is 2. Then we could use the projection to get the feasible $\bar{x}_1$ value. At the lower boundary line, if the coefficient is greater than 0, then $\bar{x}_1 = -1$, otherwise, $\bar{x}_1 = 1$. When projecting at feasible point, we have the following two situations:

1. if the coefficient is greater than 0, $\bar{x}_1 = -1 + p$,
2. otherwise, $\bar{x}_1 = 1 - p$.

Therefore, we get $\bar{x}_1 = 1 - p = 1 - \frac{2.53}{6.32} = 1 - 0.4 = 0.6$.

Similarly, we could apply the same method to get other $\bar{x}_i$’s value. #

Since the difference with coordinate descent method is only the starting point and the starting point generated from projection coordinate descent method is better compared with midpoint used in coordinate descent method, the projection coordinate descent method is expected to be faster than coordinate descent method. Numerical results will be presented in section 5.4 to demonstrate its effectiveness.

Since projection coordinate descent method is generated from the bounding step (Step 3) in section 5.2, we can conclude an overall algorithm as follows:

Step 1: for $i=1, 2...n$, normalize variable $x_i$’s domain into $[-1, 1]$, get all new variables $\bar{x}_i$ in terms of $x_i$. Initialize ordered pair $(a, b)$.

Step 2: generate the new system of two equations in terms of $\{\bar{x}_i\}$, substitute the right hand side of the new equations’ value by $\alpha, \beta$. #
5.4 Data Comparison

In this section, we compared the above three methods using four parameters, f_{best}, CPU time, iterations, and feasibility. In order to demonstrate the effectiveness of these methods, we applied them in different examples from 9901-9925 (see appendix). To better validate the result, we presented these examples with 4 dimensions, 10 dimensions, 40 dimensions, and 100 dimensions. Detailed results will be represented with easy understandable graphs.

Note: for the feasibility parameter, we compare it with the standard Hansen method with constraints. When running the standard Hansen method with constraints, we can only find solutions for a few cases. For other cases, we update the f_{best} with the f_{best} use that was introduced in chapter 2. Therefore, the feasibility parameter for the standard Hansen method is extremely bad.
Starting with 4 dimensional cases, for the $f_{\text{best}}$ parameter in table 5.1, we observed that the $f_{\text{best}}$ generated by the standard Hansen method is relatively large compared with other three methods. We didn’t include the result for the standard Hansen method in our figures because of its poor feasibility. This applies to other dimensions. For the following three figures, we observed good $f_{\text{best}}$, although for coordinate descent method there is one case with the $f_{\text{best}}$ less than 0.002. For the iteration and CPU time parameters, we observed that projection coordinate descent method and cutting line are better than coordinate descent method.

![Figure 5.10 $f_{\text{best}}$ comparison in 4 dimensions](image)

Figure 5.10 $f_{\text{best}}$ comparison in 4 dimensions
Figure 5.11 Iteration comparison in 4 dimensions

Figure 5.12 CPU Time comparison in 4 dimensions
<table>
<thead>
<tr>
<th>#</th>
<th>id</th>
<th>Unconstrain</th>
<th>Hansen</th>
<th>CD</th>
<th>PCD</th>
<th>CL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9901</td>
<td>5.03E-05</td>
<td>10827</td>
<td>5.03E-05</td>
<td>6.12E-26</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>9902</td>
<td>2.77E-05</td>
<td>9154130</td>
<td>3.39E-05</td>
<td>6.24E-09</td>
<td>7.47E-05</td>
</tr>
<tr>
<td>3</td>
<td>9903</td>
<td>9.54E-05</td>
<td>44100</td>
<td>6.71E-06</td>
<td>1.71E-28</td>
<td>7.15E-05</td>
</tr>
<tr>
<td>4</td>
<td>9904</td>
<td>8.39E-05</td>
<td>6.10352E-05</td>
<td>4.96E-05</td>
<td>4.27E-05</td>
<td>9.92E-05</td>
</tr>
<tr>
<td>5</td>
<td>9905</td>
<td>3.81E-05</td>
<td>1.01328E-05</td>
<td>8.58E-05</td>
<td>6.86E-05</td>
<td>4.36E-05</td>
</tr>
<tr>
<td>6</td>
<td>9906</td>
<td>0</td>
<td>160000</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>9907</td>
<td>1.14E-05</td>
<td>2.50778E-05</td>
<td>5.68E-06</td>
<td>0</td>
<td>8.68E-05</td>
</tr>
<tr>
<td>8</td>
<td>9908</td>
<td>-2.85E-08</td>
<td>-2.8459E-08</td>
<td>-2.85E-08</td>
<td>-2.85E-08</td>
<td>-2.85E-08</td>
</tr>
<tr>
<td>9</td>
<td>9909</td>
<td>4.49E-05</td>
<td>359.025</td>
<td>4.49E-05</td>
<td>1.23E-05</td>
<td>4.87E-05</td>
</tr>
<tr>
<td>10</td>
<td>9910</td>
<td>4.68E-05</td>
<td>361.419</td>
<td>4.82E-05</td>
<td>1.91E-05</td>
<td>8.05E-05</td>
</tr>
<tr>
<td>11</td>
<td>9911</td>
<td>5.76E-05</td>
<td>361.419</td>
<td>5.44E-05</td>
<td>0</td>
<td>6.90E-05</td>
</tr>
<tr>
<td>12</td>
<td>9912</td>
<td>1.46E-05</td>
<td>144</td>
<td>1.89E-03</td>
<td>7.09E-09</td>
<td>2.44E-05</td>
</tr>
<tr>
<td>13</td>
<td>9913</td>
<td>2.26E-06</td>
<td>12</td>
<td>0.00150806</td>
<td>4.09429E-06</td>
<td>8.47409E-05</td>
</tr>
<tr>
<td>14</td>
<td>9914</td>
<td>3.70E-05</td>
<td>4102970</td>
<td>2.55E-05</td>
<td>2.55E-05</td>
<td>1.69E-05</td>
</tr>
<tr>
<td>15</td>
<td>9915</td>
<td>1.53E-05</td>
<td>39204</td>
<td>9.11E-05</td>
<td>8.66E-05</td>
<td>1.37E-05</td>
</tr>
<tr>
<td>16</td>
<td>9916</td>
<td>7.61E-05</td>
<td>1.19208</td>
<td>7.61E-05</td>
<td>9.88E-05</td>
<td>5.80E-05</td>
</tr>
<tr>
<td>17</td>
<td>9917</td>
<td>3.82E-05</td>
<td>2430</td>
<td>7.65E-06</td>
<td>1.15E-05</td>
<td>2.30E-05</td>
</tr>
<tr>
<td>18</td>
<td>9918</td>
<td>3.05E-05</td>
<td>99</td>
<td>3.05E-05</td>
<td>8.46E-05</td>
<td>3.05E-05</td>
</tr>
<tr>
<td>19</td>
<td>9919</td>
<td>7.22E-05</td>
<td>30000000000</td>
<td>7.22E-05</td>
<td>2.88E-06</td>
<td>3.79E-05</td>
</tr>
<tr>
<td>20</td>
<td>9920</td>
<td>3.67E-05</td>
<td>2.99002</td>
<td>2.86E-05</td>
<td>3.38E-05</td>
<td>3.20E-05</td>
</tr>
<tr>
<td>21</td>
<td>9921</td>
<td>8.77E-05</td>
<td>22.3098</td>
<td>8.50E-05</td>
<td>3.98E-05</td>
<td>9.73E-05</td>
</tr>
<tr>
<td>22</td>
<td>9922</td>
<td>9.15E-05</td>
<td>360.801</td>
<td>1.12E-05</td>
<td>1.97E-05</td>
<td>7.89E-05</td>
</tr>
<tr>
<td>23</td>
<td>9923</td>
<td>3.06E-05</td>
<td>359.789</td>
<td>2.03E-05</td>
<td>1.24E-05</td>
<td>2.79E-05</td>
</tr>
<tr>
<td>24</td>
<td>9924</td>
<td>3.06E-05</td>
<td>363.195</td>
<td>2.64E-05</td>
<td>3.98E-05</td>
<td>2.53E-05</td>
</tr>
<tr>
<td>25</td>
<td>9925</td>
<td>6.06E-06</td>
<td>363.195</td>
<td>8.79E-05</td>
<td>7.95E-05</td>
<td>4.01E-06</td>
</tr>
</tbody>
</table>

Table 5.1 Detailed $f_{best}$ data for four methods and the unconstrained case in 4 dimensions
<table>
<thead>
<tr>
<th>#</th>
<th>id</th>
<th>Hansen</th>
<th>CD</th>
<th>PCD</th>
<th>CL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9901</td>
<td>0.031</td>
<td>4034.263</td>
<td>0.094</td>
<td>0.016</td>
</tr>
<tr>
<td>2</td>
<td>9902</td>
<td>0.015</td>
<td>559.638</td>
<td>0.046</td>
<td>0.062</td>
</tr>
<tr>
<td>3</td>
<td>9903</td>
<td>0.047</td>
<td>494.274</td>
<td>0.063</td>
<td>0.016</td>
</tr>
<tr>
<td>4</td>
<td>9904</td>
<td>0.047</td>
<td>0.047</td>
<td>0.047</td>
<td>0.125</td>
</tr>
<tr>
<td>5</td>
<td>9905</td>
<td>0.031</td>
<td>0.078</td>
<td>0.031</td>
<td>0.047</td>
</tr>
<tr>
<td>6</td>
<td>9906</td>
<td>0.032</td>
<td>2665.699</td>
<td>0.002</td>
<td>0.046</td>
</tr>
<tr>
<td>7</td>
<td>9907</td>
<td>0.046</td>
<td>1.154</td>
<td>0.016</td>
<td>0.047</td>
</tr>
<tr>
<td>8</td>
<td>9908</td>
<td>0.047</td>
<td>0.015</td>
<td>0.015</td>
<td>0.093</td>
</tr>
<tr>
<td>9</td>
<td>9909</td>
<td>0.062</td>
<td>332.36</td>
<td>0.047</td>
<td>0.312</td>
</tr>
<tr>
<td>10</td>
<td>9910</td>
<td>0.062</td>
<td>325.559</td>
<td>0.015</td>
<td>0.515</td>
</tr>
<tr>
<td>11</td>
<td>9911</td>
<td>0.031</td>
<td>642.303</td>
<td>0.031</td>
<td>0.203</td>
</tr>
<tr>
<td>12</td>
<td>9912</td>
<td>0.296</td>
<td>5385.919</td>
<td>56.092</td>
<td>0.031</td>
</tr>
<tr>
<td>13</td>
<td>9913</td>
<td>142.262</td>
<td>5392.409</td>
<td>12.084</td>
<td>0.047</td>
</tr>
<tr>
<td>14</td>
<td>9914</td>
<td>0.109</td>
<td>1459.607</td>
<td>0.078</td>
<td>0.187</td>
</tr>
<tr>
<td>15</td>
<td>9915</td>
<td>0.062</td>
<td>1321.859</td>
<td>0.327</td>
<td>0.125</td>
</tr>
<tr>
<td>16</td>
<td>9916</td>
<td>0.172</td>
<td>1.155</td>
<td>0.593</td>
<td>0.249</td>
</tr>
<tr>
<td>17</td>
<td>9917</td>
<td>8.18</td>
<td>1632.036</td>
<td>12.084</td>
<td>0.344</td>
</tr>
<tr>
<td>18</td>
<td>9918</td>
<td>0.188</td>
<td>240.289</td>
<td>0.655</td>
<td>0.141</td>
</tr>
<tr>
<td>19</td>
<td>9919</td>
<td>0.64</td>
<td>5250.869</td>
<td>2.373</td>
<td>0.249</td>
</tr>
<tr>
<td>20</td>
<td>9920</td>
<td>0.156</td>
<td>1639.523</td>
<td>0.172</td>
<td>0.281</td>
</tr>
<tr>
<td>21</td>
<td>9921</td>
<td>0.219</td>
<td>1735.839</td>
<td>0.53</td>
<td>0.109</td>
</tr>
<tr>
<td>22</td>
<td>9922</td>
<td>0.14</td>
<td>1259.037</td>
<td>0.437</td>
<td>0.016</td>
</tr>
<tr>
<td>23</td>
<td>9923</td>
<td>0.125</td>
<td>1868.923</td>
<td>0.546</td>
<td>0.234</td>
</tr>
<tr>
<td>24</td>
<td>9924</td>
<td>0.156</td>
<td>809.614</td>
<td>0.358</td>
<td>0.249</td>
</tr>
<tr>
<td>25</td>
<td>9925</td>
<td>0.218</td>
<td>1459.467</td>
<td>0.422</td>
<td>0.814</td>
</tr>
</tbody>
</table>

Table 5.2 Detailed CPU Time data for four methods and the unconstrained case in 4 dimensions
<table>
<thead>
<tr>
<th>#</th>
<th>id</th>
<th>Hansen</th>
<th>CD</th>
<th>PCD</th>
<th>CL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9901</td>
<td>72</td>
<td>150000</td>
<td>227</td>
<td>13</td>
</tr>
<tr>
<td>2</td>
<td>9902</td>
<td>44</td>
<td>150000</td>
<td>42</td>
<td>41</td>
</tr>
<tr>
<td>3</td>
<td>9903</td>
<td>48</td>
<td>150000</td>
<td>64</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>9904</td>
<td>70</td>
<td>76</td>
<td>71</td>
<td>177</td>
</tr>
<tr>
<td>5</td>
<td>9905</td>
<td>72</td>
<td>217</td>
<td>75</td>
<td>70</td>
</tr>
<tr>
<td>6</td>
<td>9906</td>
<td>28</td>
<td>150000</td>
<td>29</td>
<td>20</td>
</tr>
<tr>
<td>7</td>
<td>9907</td>
<td>52</td>
<td>3895</td>
<td>52</td>
<td>37</td>
</tr>
<tr>
<td>8</td>
<td>9908</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>134</td>
</tr>
<tr>
<td>9</td>
<td>9909</td>
<td>60</td>
<td>150000</td>
<td>60</td>
<td>723</td>
</tr>
<tr>
<td>10</td>
<td>9910</td>
<td>52</td>
<td>150000</td>
<td>53</td>
<td>858</td>
</tr>
<tr>
<td>11</td>
<td>9911</td>
<td>66</td>
<td>150000</td>
<td>65</td>
<td>347</td>
</tr>
<tr>
<td>12</td>
<td>9912</td>
<td>903</td>
<td>144040</td>
<td>15000</td>
<td>4</td>
</tr>
<tr>
<td>13</td>
<td>9913</td>
<td>15000</td>
<td>150000</td>
<td>15000</td>
<td>54</td>
</tr>
<tr>
<td>14</td>
<td>9914</td>
<td>448</td>
<td>150000</td>
<td>447</td>
<td>259</td>
</tr>
<tr>
<td>15</td>
<td>9915</td>
<td>640</td>
<td>150000</td>
<td>677</td>
<td>149</td>
</tr>
<tr>
<td>16</td>
<td>9916</td>
<td>1024</td>
<td>3700</td>
<td>1039</td>
<td>485</td>
</tr>
<tr>
<td>17</td>
<td>9917</td>
<td>6337</td>
<td>150000</td>
<td>6484</td>
<td>735</td>
</tr>
<tr>
<td>18</td>
<td>9918</td>
<td>1024</td>
<td>150000</td>
<td>1039</td>
<td>172</td>
</tr>
<tr>
<td>19</td>
<td>9919</td>
<td>1921</td>
<td>150000</td>
<td>2097</td>
<td>429</td>
</tr>
<tr>
<td>20</td>
<td>9920</td>
<td>640</td>
<td>150000</td>
<td>655</td>
<td>445</td>
</tr>
<tr>
<td>21</td>
<td>9921</td>
<td>1072</td>
<td>150000</td>
<td>1148</td>
<td>137</td>
</tr>
<tr>
<td>22</td>
<td>9922</td>
<td>512</td>
<td>150000</td>
<td>527</td>
<td>3</td>
</tr>
<tr>
<td>23</td>
<td>9923</td>
<td>512</td>
<td>150000</td>
<td>527</td>
<td>339</td>
</tr>
<tr>
<td>24</td>
<td>9924</td>
<td>512</td>
<td>150000</td>
<td>591</td>
<td>356</td>
</tr>
<tr>
<td>25</td>
<td>9925</td>
<td>832</td>
<td>150000</td>
<td>751</td>
<td>10492</td>
</tr>
</tbody>
</table>

Table 5.3 Detailed iteration data for four methods and the unconstraint case in 4 dimensions
Table 5.4 Feasibility for four methods in 4 dimensions

For examples 9912, 9913, and 9917, because of their sensitivity and the data we collected for these examples in each method are relatively high, we excluded these 3 examples in 10 dimensions, 40 dimensions and 100 dimensions. For 100 dimensions, we didn’t include example 9921 for the same reason.

Similarly to 4 dimensions’ results, for $f_{\text{best}}$ parameter in 10 dimensions in table 5.5, we found that the $f_{\text{best}}$ results generated by the standard Hansen method are relatively large compared
with other three methods. For the following three figures, we observed good $f_{\text{best}}$ results, although for coordinate descent method there is one case with the $f_{\text{best}}$ being around 1.4, which means the coordinate descent method’s convergence is poor for this case. For the iteration and CPU time parameters, we found that projection coordinate descent method and cutting line are better than the coordinate descent method, and the cutting line method seems to be the best.

**Figure 5.13 $f_{\text{best}}$ comparison in 10 dimensions**

![fbest comparison](image)

**Figure 5.14 Iteration comparison in 10 dimensions**

![Iterations](image)
Figure 5.15 CPU Time comparison in 10 dimensions
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Hansen</th>
<th>CD</th>
<th>PCD</th>
<th>CL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9901</td>
<td>8.43E-05</td>
<td>9180910000</td>
<td>4.12E-05</td>
<td>1.84E-25</td>
</tr>
<tr>
<td>2</td>
<td>9902</td>
<td>8.85E-05</td>
<td>8373490000</td>
<td>4.06E-05</td>
<td>8.32E-06</td>
</tr>
<tr>
<td>3</td>
<td>9903</td>
<td>6.71E-05</td>
<td>110250</td>
<td>5.81E-05</td>
<td>4.04E-28</td>
</tr>
<tr>
<td>4</td>
<td>9904</td>
<td>7.25E-05</td>
<td>3.05176E-05</td>
<td>3.81E-05</td>
<td>7.75E-05</td>
</tr>
<tr>
<td>5</td>
<td>9905</td>
<td>3.81E-05</td>
<td>100</td>
<td>4.77E-05</td>
<td>1.22E-05</td>
</tr>
<tr>
<td>6</td>
<td>9906</td>
<td>0</td>
<td>400000</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>9907</td>
<td>2.84E-05</td>
<td>574.544</td>
<td>6.73E-05</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>9908</td>
<td>-2.85E-08</td>
<td>-2.8459E-08</td>
<td>-2.85E-08</td>
<td>-2.85E-08</td>
</tr>
<tr>
<td>9</td>
<td>9909</td>
<td>9.17E-05</td>
<td>902</td>
<td>8.98E-05</td>
<td>2.39E-05</td>
</tr>
<tr>
<td>10</td>
<td>9910</td>
<td>6.60E-05</td>
<td>910.986</td>
<td>7.70E-06</td>
<td>1.78E-15</td>
</tr>
<tr>
<td>11</td>
<td>9911</td>
<td>7.76E-05</td>
<td>910.986</td>
<td>4.39E-05</td>
<td>1.78E-15</td>
</tr>
<tr>
<td>12</td>
<td>9914</td>
<td>5.83E-05</td>
<td>4064920000</td>
<td>7.76E-05</td>
<td>0.0567252</td>
</tr>
<tr>
<td>13</td>
<td>9915</td>
<td>8.51E-05</td>
<td>100410</td>
<td>8.94E-05</td>
<td>6.51E-05</td>
</tr>
<tr>
<td>14</td>
<td>9916</td>
<td>9.90E-05</td>
<td>1162320000</td>
<td>9.90E-05</td>
<td>0.0001006</td>
</tr>
<tr>
<td>15</td>
<td>9918</td>
<td>7.25E-05</td>
<td>101</td>
<td>7.25E-05</td>
<td>9.33E-05</td>
</tr>
<tr>
<td>16</td>
<td>9919</td>
<td>6.83E-05</td>
<td>9120610000</td>
<td>5.06E-05</td>
<td>8.74E-05</td>
</tr>
<tr>
<td>17</td>
<td>9920</td>
<td>0.168964</td>
<td>492.624</td>
<td>0.136297</td>
<td>3.20E-05</td>
</tr>
<tr>
<td>18</td>
<td>9921</td>
<td>2.21078</td>
<td>22.3186</td>
<td>1.35741</td>
<td>0.0969103</td>
</tr>
<tr>
<td>19</td>
<td>9922</td>
<td>6.85E-05</td>
<td>902.602</td>
<td>6.79E-05</td>
<td>8.57E-05</td>
</tr>
<tr>
<td>20</td>
<td>9923</td>
<td>8.51E-05</td>
<td>901.602</td>
<td>9.45E-05</td>
<td>8.43E-05</td>
</tr>
<tr>
<td>21</td>
<td>9924</td>
<td>8.51E-05</td>
<td>911.589</td>
<td>7.22E-05</td>
<td>6.37E-05</td>
</tr>
<tr>
<td>22</td>
<td>9925</td>
<td>5.61E-05</td>
<td>911.589</td>
<td>7.70E-05</td>
<td>5.68E-05</td>
</tr>
</tbody>
</table>

Table 5.5 Detailed $f_{\text{best}}$ data for four methods and the unconstrained case in 10 dimensions
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Hansen</th>
<th>Unconstraint</th>
<th>CD</th>
<th>PCD</th>
<th>CL</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9901</td>
<td>0.062</td>
<td>1970.261</td>
<td>0.296</td>
<td>0.093</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>9902</td>
<td>0.094</td>
<td>2146.09</td>
<td>0.11</td>
<td>0.484</td>
<td>0.171</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>9903</td>
<td>0.047</td>
<td>2099.477</td>
<td>0.109</td>
<td>0.078</td>
<td>0.141</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>9904</td>
<td>0.093</td>
<td>0.094</td>
<td>0.094</td>
<td>1.092</td>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>9905</td>
<td>0.078</td>
<td>2704.777</td>
<td>0.256</td>
<td>0.951</td>
<td>0.249</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>9906</td>
<td>0.063</td>
<td>5396.059</td>
<td>0.093</td>
<td>0.312</td>
<td>0.562</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>9907</td>
<td>0.109</td>
<td>1753.732</td>
<td>0.11</td>
<td>0.172</td>
<td>0.187</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>9908</td>
<td>0.047</td>
<td>0.047</td>
<td>0.078</td>
<td>0.234</td>
<td>0.453</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9909</td>
<td>0.062</td>
<td>919.828</td>
<td>0.187</td>
<td>0.64</td>
<td>0.171</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>9910</td>
<td>0.078</td>
<td>1328.942</td>
<td>0.078</td>
<td>0.421</td>
<td>0.125</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>9911</td>
<td>0.109</td>
<td>3702.699</td>
<td>0.156</td>
<td>0.312</td>
<td>0.234</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>9914</td>
<td>0.452</td>
<td>2263.044</td>
<td>0.562</td>
<td>19.594</td>
<td>0.609</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>9915</td>
<td>0.39</td>
<td>2005.549</td>
<td>4.466</td>
<td>3.463</td>
<td>0.827</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>9916</td>
<td>0.858</td>
<td>2998.557</td>
<td>4.134</td>
<td>2.809</td>
<td>0.904</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>9918</td>
<td>0.639</td>
<td>5389.585</td>
<td>3.37</td>
<td>5.085</td>
<td>0.873</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>9919</td>
<td>6.592</td>
<td>2332.98</td>
<td>13.525</td>
<td>13.26</td>
<td>3.775</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>9920</td>
<td>32.559</td>
<td>3324.319</td>
<td>41.902</td>
<td>21.31</td>
<td>9.875</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>9921</td>
<td>29.422</td>
<td>3237.348</td>
<td>64.257</td>
<td>48.969</td>
<td>33.29</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>9922</td>
<td>0.484</td>
<td>2243.653</td>
<td>2.418</td>
<td>3.869</td>
<td>0.92</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>9923</td>
<td>0.437</td>
<td>2077.497</td>
<td>1.763</td>
<td>4.149</td>
<td>1.248</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>9924</td>
<td>0.468</td>
<td>2061.46</td>
<td>1.388</td>
<td>4.743</td>
<td>4.326</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>9925</td>
<td>7.077</td>
<td>2705.838</td>
<td>2.59</td>
<td>5.663</td>
<td>1.747</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.6 Detailed CPU Time data for four methods and the unconstrained case in 10 dimensions
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Hansen</th>
<th>CD</th>
<th>PCD</th>
<th>CL</th>
<th>Unconstraint</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9901</td>
<td>213</td>
<td>150000</td>
<td>657</td>
<td>10</td>
<td>418</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>9902</td>
<td>115</td>
<td>150000</td>
<td>112</td>
<td>90</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>9903</td>
<td>139</td>
<td>150000</td>
<td>141</td>
<td>10</td>
<td>139</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>9904</td>
<td>179</td>
<td></td>
<td>204</td>
<td>180</td>
<td>178</td>
<td>187</td>
</tr>
<tr>
<td>5</td>
<td>9905</td>
<td>180</td>
<td>150000</td>
<td>202</td>
<td>180</td>
<td>202</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>9906</td>
<td>70</td>
<td>107720</td>
<td>133</td>
<td>60</td>
<td>471</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>9907</td>
<td>130</td>
<td>150000</td>
<td>132</td>
<td>30</td>
<td>140</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>9908</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>323</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9909</td>
<td>149</td>
<td>150000</td>
<td>152</td>
<td>130</td>
<td>157</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>9910</td>
<td>130</td>
<td>150000</td>
<td>157</td>
<td>70</td>
<td>142</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>9911</td>
<td>179</td>
<td>150000</td>
<td>181</td>
<td>70</td>
<td>188</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>9914</td>
<td>1772</td>
<td>150000</td>
<td>1483</td>
<td></td>
<td>4690</td>
<td>418</td>
</tr>
<tr>
<td>13</td>
<td>9915</td>
<td>1688</td>
<td>150000</td>
<td>1772</td>
<td>714</td>
<td>715</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>9916</td>
<td>2652</td>
<td>150000</td>
<td>2683</td>
<td>556</td>
<td>568</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>9918</td>
<td>2584</td>
<td>127010</td>
<td>2744</td>
<td>790</td>
<td>802</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>9919</td>
<td>8490</td>
<td>150000</td>
<td>8808</td>
<td>2886</td>
<td>2906</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>9920</td>
<td>15000</td>
<td>150000</td>
<td>15000</td>
<td>3930</td>
<td>3871</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>9921</td>
<td>15000</td>
<td>150000</td>
<td>15000</td>
<td>7820</td>
<td>13330</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>9922</td>
<td>1758</td>
<td>150000</td>
<td>1693</td>
<td>867</td>
<td>877</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>9923</td>
<td>1390</td>
<td>150000</td>
<td>1453</td>
<td>907</td>
<td>915</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>9924</td>
<td>1390</td>
<td>150000</td>
<td>1775</td>
<td>937</td>
<td>957</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>9925</td>
<td>2110</td>
<td>150000</td>
<td>2493</td>
<td>1011</td>
<td>1040</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.7 Detailed iteration data for four methods and the unconstrained case in 10 dimensions
Table 5.8 Feasibility for four methods in 10 dimensions

For 40 dimensions, the computation becomes very time consuming because we have more variables. Except the feasibility parameter, all other three parameters became worse for some cases such as case 9906, 9919, and 9920. However, if we compare these results with the standard Hansen method, we still found them to be much better. The CPU time parameter is difficult to compare since in C++ programming, the machine might get overheated because of constant computation, which will incur abnormal long CPU time. However, we can still demonstrate the effectiveness of the three method using the iteration parameter. In figure 5.17, all these three methods almost overlapped except for a few cases.
Figure 5.16 $f_{\text{best}}$ comparison in 40 dimensions

Figure 5.17 Iteration comparison in 40 dimensions
Figure 5.18 CPU Time comparison in 40 dimensions
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Unconstraint</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ID</td>
<td>9.78E-05</td>
<td>3.97839E+11</td>
</tr>
<tr>
<td>2</td>
<td>9901</td>
<td>8.14E-05</td>
<td>4.1372E+14</td>
</tr>
<tr>
<td>3</td>
<td>9902</td>
<td>5.96E-05</td>
<td>0.000089407</td>
</tr>
<tr>
<td>4</td>
<td>9903</td>
<td>0.000100613</td>
<td>10000</td>
</tr>
<tr>
<td>5</td>
<td>9904</td>
<td>3.81E-05</td>
<td>100</td>
</tr>
<tr>
<td>6</td>
<td>9905</td>
<td>0</td>
<td>1600000</td>
</tr>
<tr>
<td>7</td>
<td>9906</td>
<td>9.63E-05</td>
<td>2298.18</td>
</tr>
<tr>
<td>8</td>
<td>9907</td>
<td>-2.85E-08</td>
<td>-2.85E-08</td>
</tr>
<tr>
<td>9</td>
<td>9908</td>
<td>9.98E-05</td>
<td>3602</td>
</tr>
<tr>
<td>10</td>
<td>9909</td>
<td>9.72E-05</td>
<td>3643.94</td>
</tr>
<tr>
<td>11</td>
<td>9910</td>
<td>9.93E-05</td>
<td>3643.94</td>
</tr>
<tr>
<td>12</td>
<td>9911</td>
<td>2.77E-05</td>
<td>5370000</td>
</tr>
<tr>
<td>13</td>
<td>9914</td>
<td>9.72E-05</td>
<td>406440</td>
</tr>
<tr>
<td>14</td>
<td>9915</td>
<td>0.000100321</td>
<td>65600</td>
</tr>
<tr>
<td>15</td>
<td>9916</td>
<td>7.25E-05</td>
<td>101</td>
</tr>
<tr>
<td>16</td>
<td>9918</td>
<td>9.95E-05</td>
<td>3.97236E+11</td>
</tr>
<tr>
<td>17</td>
<td>9919</td>
<td>2.06637</td>
<td>2216.26</td>
</tr>
<tr>
<td>18</td>
<td>9920</td>
<td>1.1153</td>
<td>22.3221</td>
</tr>
<tr>
<td>19</td>
<td>9921</td>
<td>7.15E-05</td>
<td>3611.61</td>
</tr>
<tr>
<td>20</td>
<td>9922</td>
<td>9.72E-05</td>
<td>3610.61</td>
</tr>
<tr>
<td>21</td>
<td>9923</td>
<td>9.72E-05</td>
<td>3653.55</td>
</tr>
<tr>
<td>22</td>
<td>9924</td>
<td>8.97E-05</td>
<td>3653.55</td>
</tr>
</tbody>
</table>

Table 5.9 Detailed $f_{\text{best}}$ data for four methods and the unconstrained case in 40 dimensions
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Unconstraint</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Hansen</td>
<td>CD</td>
</tr>
<tr>
<td>1</td>
<td>9901</td>
<td>0.686</td>
<td>1834.806</td>
</tr>
<tr>
<td>2</td>
<td>9902</td>
<td>0.203</td>
<td>4422.442</td>
</tr>
<tr>
<td>3</td>
<td>9903</td>
<td>0.171</td>
<td>0.343</td>
</tr>
<tr>
<td>4</td>
<td>9904</td>
<td>0.39</td>
<td>3767.206</td>
</tr>
<tr>
<td>5</td>
<td>9905</td>
<td>0.218</td>
<td>5401.02</td>
</tr>
<tr>
<td>6</td>
<td>9906</td>
<td>0.156</td>
<td>5399.927</td>
</tr>
<tr>
<td>7</td>
<td>9907</td>
<td>0.343</td>
<td>5389.241</td>
</tr>
<tr>
<td>8</td>
<td>9908</td>
<td>0.109</td>
<td>0.156</td>
</tr>
<tr>
<td>9</td>
<td>9909</td>
<td>0.343</td>
<td>4423.206</td>
</tr>
<tr>
<td>10</td>
<td>9910</td>
<td>0.328</td>
<td>2484.316</td>
</tr>
<tr>
<td>11</td>
<td>9911</td>
<td>1.81</td>
<td>5401.768</td>
</tr>
<tr>
<td>12</td>
<td>9914</td>
<td>8.955</td>
<td>1595.313</td>
</tr>
<tr>
<td>13</td>
<td>9915</td>
<td>5.304</td>
<td>1844.307</td>
</tr>
<tr>
<td>14</td>
<td>9916</td>
<td>18.112</td>
<td>5069.19</td>
</tr>
<tr>
<td>15</td>
<td>9918</td>
<td>4.441</td>
<td>5406.994</td>
</tr>
<tr>
<td>16</td>
<td>9919</td>
<td>151.041</td>
<td>1698.648</td>
</tr>
<tr>
<td>17</td>
<td>9920</td>
<td>5394.155</td>
<td>1638.853</td>
</tr>
<tr>
<td>18</td>
<td>9921</td>
<td>5394.982</td>
<td>1415.038</td>
</tr>
<tr>
<td>19</td>
<td>9922</td>
<td>8.642</td>
<td>1508.966</td>
</tr>
<tr>
<td>20</td>
<td>9923</td>
<td>6.911</td>
<td>1812.809</td>
</tr>
<tr>
<td>21</td>
<td>9924</td>
<td>6.911</td>
<td>1857.629</td>
</tr>
<tr>
<td>22</td>
<td>9925</td>
<td>37.013</td>
<td>1619.259</td>
</tr>
</tbody>
</table>

Table 5.10 Detailed CPU Time data for four methods and the unconstrained case in 40 dimensions
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Hansen</th>
<th>CD</th>
<th>PCD</th>
<th>CL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9901</td>
<td>968</td>
<td>150000</td>
<td>368</td>
<td>40</td>
</tr>
<tr>
<td>2</td>
<td>9902</td>
<td>465</td>
<td>150000</td>
<td>560</td>
<td>120</td>
</tr>
<tr>
<td>3</td>
<td>9903</td>
<td>560</td>
<td>576</td>
<td>576</td>
<td>40</td>
</tr>
<tr>
<td>4</td>
<td>9904</td>
<td>821</td>
<td>150000</td>
<td>63359</td>
<td>719</td>
</tr>
<tr>
<td>5</td>
<td>9905</td>
<td>720</td>
<td>91070</td>
<td>819</td>
<td>720</td>
</tr>
<tr>
<td>6</td>
<td>9906</td>
<td>280</td>
<td>89510</td>
<td>92520</td>
<td>800</td>
</tr>
<tr>
<td>7</td>
<td>9907</td>
<td>599</td>
<td>126810</td>
<td>598</td>
<td>200</td>
</tr>
<tr>
<td>8</td>
<td>9908</td>
<td>592</td>
<td>200</td>
<td>200</td>
<td>200</td>
</tr>
<tr>
<td>9</td>
<td>9909</td>
<td>599</td>
<td>150000</td>
<td>626</td>
<td>600</td>
</tr>
<tr>
<td>10</td>
<td>9910</td>
<td>520</td>
<td>150000</td>
<td>628</td>
<td>280</td>
</tr>
<tr>
<td>11</td>
<td>9911</td>
<td>839</td>
<td>87920</td>
<td>840</td>
<td>280</td>
</tr>
<tr>
<td>12</td>
<td>9914</td>
<td>7448</td>
<td>150000</td>
<td>2563</td>
<td>961</td>
</tr>
<tr>
<td>13</td>
<td>9915</td>
<td>7552</td>
<td>150000</td>
<td>7551</td>
<td>4198</td>
</tr>
<tr>
<td>14</td>
<td>9916</td>
<td>12040</td>
<td>150000</td>
<td>6881</td>
<td>3650</td>
</tr>
<tr>
<td>15</td>
<td>9918</td>
<td>10384</td>
<td>90080</td>
<td>11024</td>
<td>4402</td>
</tr>
<tr>
<td>16</td>
<td>9919</td>
<td>36421</td>
<td>150000</td>
<td>38034</td>
<td>20036</td>
</tr>
<tr>
<td>17</td>
<td>9920</td>
<td>130500</td>
<td>150000</td>
<td>94820</td>
<td>75000</td>
</tr>
<tr>
<td>18</td>
<td>9921</td>
<td>129150</td>
<td>150000</td>
<td>6520</td>
<td>3210</td>
</tr>
<tr>
<td>19</td>
<td>9922</td>
<td>7188</td>
<td>150000</td>
<td>7011</td>
<td>4408</td>
</tr>
<tr>
<td>20</td>
<td>9923</td>
<td>5844</td>
<td>150000</td>
<td>6963</td>
<td>4973</td>
</tr>
<tr>
<td>21</td>
<td>9924</td>
<td>5844</td>
<td>150000</td>
<td>7075</td>
<td>5192</td>
</tr>
<tr>
<td>22</td>
<td>9925</td>
<td>11668</td>
<td>150000</td>
<td>11715</td>
<td>5604</td>
</tr>
</tbody>
</table>

Table 5.11 Detailed iteration data for four methods and the unconstrained case in 40 dimensions
For 100 dimensions, computation took the longest time since the dimensions for these cases are the largest. We excluded the example 9921 in 100 dimensions. If we compare these results with the standard Hansen method, we still found them to be much better. The CPU time parameter is difficult to compare for the same reason we have for 40 dimensions. However, the iteration parameter can still demonstrate the effectiveness of these three methods. In figure 5.20, we observed that the coordinate descent method took more iterations than the projection coordinate descent method and the cutting line method.
Figure 5.19 f_{best} comparison in 100 dimensions

Figure 5.20 Iteration comparison in 100 dimensions
Figure 5.21 CPU Time comparison in 100 dimensions
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Unconstraint</th>
<th>Hansen</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9901</td>
<td>0.000100066</td>
<td>4906.06</td>
<td>9.63E-05</td>
</tr>
<tr>
<td>2</td>
<td>9902</td>
<td>1.21E-05</td>
<td>5.95136E+17</td>
<td>2.13E-05</td>
</tr>
<tr>
<td>3</td>
<td>9903</td>
<td>0.000100583</td>
<td>1.39698E-05</td>
<td>9.31E-05</td>
</tr>
<tr>
<td>4</td>
<td>9904</td>
<td>9.92E-05</td>
<td>10000000000</td>
<td>0.000519</td>
</tr>
<tr>
<td>5</td>
<td>9905</td>
<td>3.81E-05</td>
<td>100</td>
<td>9.92E-05</td>
</tr>
<tr>
<td>6</td>
<td>9906</td>
<td>0</td>
<td>4000000</td>
<td>27</td>
</tr>
<tr>
<td>7</td>
<td>9907</td>
<td>0.000100229</td>
<td>5745.44</td>
<td>9.17E-05</td>
</tr>
<tr>
<td>8</td>
<td>9908</td>
<td>-2.85E-08</td>
<td>-2.8459E-08</td>
<td>-2.8E-08</td>
</tr>
<tr>
<td>9</td>
<td>9909</td>
<td>9.44E-05</td>
<td>9002</td>
<td>8.48E-05</td>
</tr>
<tr>
<td>10</td>
<td>9910</td>
<td>0.000100895</td>
<td>9109.86</td>
<td>6.98E-06</td>
</tr>
<tr>
<td>11</td>
<td>9911</td>
<td>9.87E-05</td>
<td>9109.86</td>
<td>8.85E-05</td>
</tr>
<tr>
<td>12</td>
<td>9914</td>
<td>3.71E-05</td>
<td>21545900000</td>
<td>2.28E-05</td>
</tr>
<tr>
<td>13</td>
<td>9915</td>
<td>9.93E-05</td>
<td>1018500</td>
<td>9.24E-05</td>
</tr>
<tr>
<td>14</td>
<td>9916</td>
<td>0.0001013</td>
<td>1.09951E+12</td>
<td>0.000396</td>
</tr>
<tr>
<td>15</td>
<td>9918</td>
<td>7.25E-05</td>
<td>101</td>
<td>7.25E-05</td>
</tr>
<tr>
<td>16</td>
<td>9919</td>
<td>9.99E-05</td>
<td>1.0093E+12</td>
<td>7.07E-05</td>
</tr>
<tr>
<td>17</td>
<td>9920</td>
<td>2.06637</td>
<td>5663.52</td>
<td>1.88649</td>
</tr>
<tr>
<td>18</td>
<td>9922</td>
<td>7.35E-05</td>
<td>9029.62</td>
<td>8.67E-05</td>
</tr>
<tr>
<td>19</td>
<td>9923</td>
<td>9.88E-05</td>
<td>9028.62</td>
<td>7.39E-05</td>
</tr>
<tr>
<td>20</td>
<td>9924</td>
<td>9.88E-05</td>
<td>9137.49</td>
<td>8.99E-05</td>
</tr>
<tr>
<td>21</td>
<td>9925</td>
<td>0.000100933</td>
<td>9137.49</td>
<td>9.94E-05</td>
</tr>
</tbody>
</table>

Table 5.13 Detailed f<sub>best</sub> data for four methods and the unconstrained case in 100 dimensions
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Hansen</th>
<th>CD</th>
<th>PCD</th>
<th>CL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9901</td>
<td>3.432</td>
<td>2916.033</td>
<td>16.583</td>
<td>74.974</td>
</tr>
<tr>
<td>2</td>
<td>9902</td>
<td>1.108</td>
<td>3278.642</td>
<td>2.902</td>
<td>172.365</td>
</tr>
<tr>
<td>3</td>
<td>9903</td>
<td>1.232</td>
<td>1.981</td>
<td>12.729</td>
<td>61.449</td>
</tr>
<tr>
<td>4</td>
<td>9904</td>
<td>1.935</td>
<td>3210.423</td>
<td>3427.997</td>
<td>1099.261</td>
</tr>
<tr>
<td>5</td>
<td>9905</td>
<td>1.435</td>
<td>5403.844</td>
<td>17.597</td>
<td>985.723</td>
</tr>
<tr>
<td>6</td>
<td>9906</td>
<td>0.624</td>
<td>5407.743</td>
<td>5404.935</td>
<td>5390.069</td>
</tr>
<tr>
<td>7</td>
<td>9907</td>
<td>1.763</td>
<td>3329.655</td>
<td>3.573</td>
<td>295.965</td>
</tr>
<tr>
<td>8</td>
<td>9908</td>
<td>0.577</td>
<td>0.577</td>
<td>1.107</td>
<td>249.632</td>
</tr>
<tr>
<td>9</td>
<td>9909</td>
<td>1.716</td>
<td>2702.718</td>
<td>13.119</td>
<td>755.451</td>
</tr>
<tr>
<td>10</td>
<td>9910</td>
<td>1.887</td>
<td>1954.957</td>
<td>33.524</td>
<td>391.126</td>
</tr>
<tr>
<td>11</td>
<td>9911</td>
<td>25.85</td>
<td>5409.974</td>
<td>33.353</td>
<td>418.504</td>
</tr>
<tr>
<td>12</td>
<td>9914</td>
<td>66.644</td>
<td>1637.184</td>
<td>146.516</td>
<td>304.288</td>
</tr>
<tr>
<td>13</td>
<td>9915</td>
<td>42.947</td>
<td>1535.908</td>
<td>225.749</td>
<td>6175.815</td>
</tr>
<tr>
<td>14</td>
<td>9916</td>
<td>1589.837</td>
<td>1966.674</td>
<td>4859.353</td>
<td>5501.688</td>
</tr>
<tr>
<td>15</td>
<td>9918</td>
<td>23.525</td>
<td>5347.964</td>
<td>450.796</td>
<td>6449.955</td>
</tr>
<tr>
<td>16</td>
<td>9919</td>
<td>1268.413</td>
<td>1817.879</td>
<td>2625.855</td>
<td>9021.444</td>
</tr>
<tr>
<td>17</td>
<td>9920</td>
<td>5392.627</td>
<td>1637.261</td>
<td>5404.234</td>
<td>9023.862</td>
</tr>
<tr>
<td>18</td>
<td>9922</td>
<td>67.142</td>
<td>1524.863</td>
<td>239.243</td>
<td>8444.491</td>
</tr>
<tr>
<td>19</td>
<td>9923</td>
<td>49.187</td>
<td>2357.253</td>
<td>370.83</td>
<td>6971.216</td>
</tr>
<tr>
<td>20</td>
<td>9924</td>
<td>50.248</td>
<td>2443.428</td>
<td>203.737</td>
<td>7329.036</td>
</tr>
<tr>
<td>21</td>
<td>9925</td>
<td>476.739</td>
<td>2928.029</td>
<td>854.932</td>
<td>8161.629</td>
</tr>
</tbody>
</table>

Table 5.14 Detailed CPU Time data for four methods and the unconstrained case in 100 dimensions
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Hansen</th>
<th>CD</th>
<th>PCD</th>
<th>CL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9901</td>
<td>2573</td>
<td>15000</td>
<td>7020</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td>9902</td>
<td>1363</td>
<td>15000</td>
<td>1415</td>
<td>300</td>
</tr>
<tr>
<td>3</td>
<td>9903</td>
<td>1572</td>
<td>1640</td>
<td>1588</td>
<td>100</td>
</tr>
<tr>
<td>4</td>
<td>9904</td>
<td>2088</td>
<td>15000</td>
<td>15000</td>
<td>2082</td>
</tr>
<tr>
<td>5</td>
<td>9905</td>
<td>1800</td>
<td>90810</td>
<td>81060</td>
<td>9590</td>
</tr>
<tr>
<td>6</td>
<td>9906</td>
<td>700</td>
<td>500</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>7</td>
<td>9907</td>
<td>1495</td>
<td>15000</td>
<td>1540</td>
<td>500</td>
</tr>
<tr>
<td>8</td>
<td>9908</td>
<td>500</td>
<td>500</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>9</td>
<td>9909</td>
<td>1501</td>
<td>15000</td>
<td>1570</td>
<td>1561</td>
</tr>
<tr>
<td>10</td>
<td>9910</td>
<td>1455</td>
<td>15000</td>
<td>1574</td>
<td>700</td>
</tr>
<tr>
<td>11</td>
<td>9911</td>
<td>2373</td>
<td>84230</td>
<td>2378</td>
<td>700</td>
</tr>
<tr>
<td>12</td>
<td>9914</td>
<td>18736</td>
<td>15000</td>
<td>18783</td>
<td>600</td>
</tr>
<tr>
<td>13</td>
<td>9915</td>
<td>19376</td>
<td>15000</td>
<td>19555</td>
<td>12753</td>
</tr>
<tr>
<td>14</td>
<td>9916</td>
<td>150000</td>
<td>150000</td>
<td>150000</td>
<td>11285</td>
</tr>
<tr>
<td>15</td>
<td>9918</td>
<td>25984</td>
<td>87950</td>
<td>27584</td>
<td>13299</td>
</tr>
<tr>
<td>16</td>
<td>9919</td>
<td>94729</td>
<td>15000</td>
<td>95686</td>
<td>15900</td>
</tr>
<tr>
<td>17</td>
<td>9920</td>
<td>122250</td>
<td>15000</td>
<td>88200</td>
<td>18550</td>
</tr>
<tr>
<td>18</td>
<td>9922</td>
<td>18048</td>
<td>150000</td>
<td>17647</td>
<td>13435</td>
</tr>
<tr>
<td>19</td>
<td>9923</td>
<td>14816</td>
<td>150000</td>
<td>17615</td>
<td>14389</td>
</tr>
<tr>
<td>20</td>
<td>9924</td>
<td>14816</td>
<td>150000</td>
<td>17887</td>
<td>15047</td>
</tr>
<tr>
<td>21</td>
<td>9925</td>
<td>33488</td>
<td>150000</td>
<td>33941</td>
<td>16170</td>
</tr>
</tbody>
</table>

Table 5.15 Detailed iteration data for four methods and the unconstrained case in 100 dimensions
<table>
<thead>
<tr>
<th>#</th>
<th>ID</th>
<th>Mid</th>
<th>CD</th>
<th>PCD</th>
<th>CL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9901</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>2</td>
<td>9902</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>3</td>
<td>9903</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>4</td>
<td>9904</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>5</td>
<td>9905</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>6</td>
<td>9906</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>7</td>
<td>9907</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>8</td>
<td>9908</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>9</td>
<td>9909</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10</td>
<td>9910</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>11</td>
<td>9911</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>12</td>
<td>9914</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>13</td>
<td>9915</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>14</td>
<td>9916</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>15</td>
<td>9918</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>16</td>
<td>9919</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>17</td>
<td>9920</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>18</td>
<td>9922</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>19</td>
<td>9923</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>20</td>
<td>9924</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>21</td>
<td>9925</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

Table 5.16 Feasibility for four methods in 100 dimensions

After comparing different dimensions, we found that the feasibility parameter for the standard Hansen method is very poor (already discussed above). This means that applying the midpoint as the sample point is not always effective based on the CPU time and the iteration parameters. Compared with the projection coordinate descent method and the coordinate descent method, the cutting line method didn’t work well for a few cases. However, it has been improved overall.

In summary, the coordinate method, the cutting line method, and the projection coordinate method guarantee the feasibility for C-GOP, but the standard Hansen method cannot.
As we observed above, for lower dimensions such as four dimensions and ten dimensions, the projection coordinate descent method and the cutting line method worked well. For higher dimensions cases such as 40 dimensions and 100 dimensions, we observed that the projection coordinate descent method and the cutting line method had a better convergence. Their iterations are relatively smaller compared with the coordinate descent method, and are much smaller compared with the standard Hansen method. The CPU times can be high because of abnormal long computation caused by overheated machines. However, if the CPU time could be improved at the same time, it is also beneficial.

Although all three methods work well, the convergence of cutting line method is not verified yet. Therefore, we will work on the convergence property of cutting line method and reducing the CPU time if possible in the future.
REFERENCES


APPENDIX: LIST OF EXAMPLES

Remark: the following list of examples has their own ID. The first number means the variables’
dimension; the following two numbers are the sequence in each dimension. Such as “403” means
it has four dimensions, and it is listed in the third place in dimension “4”. If the first two numbers
are “99”, it means the example’s dimension can be various (in this dissertation, the dimensions
here are 4, 10, 40, 100). The domain for each variable is given.

Ex101. Over [2.7, 7.5],
\[ f(x) = 5.0 + \sin(x) + \sin(10x/3) + [\log(x) - 0.84x] - 0.39869259 \]

Ex102. Over [3.1, 20.4],
\[ f(x) = 2.0 + \sin(x) + \sin(2x/3) - 0.0940388 \]

Ex103. Over [−10, 10],
\[ f(x) = 6 + \sin((r_1 + 1)x + r_1) + \sin((r_2 + 1)x + r_2) + \sin((r_3 + 1)x + r_3 + \sin((r_4 
+ 1)x + r_4) + \sin((r_5 + 1)x + r_5) - 1.0899717, \]
\[ r = (1, 2, 3, 4, 5). \]

Ex104. Over [−10, 10],
\[ f(x) = 1.0 + [x + \sin(x)] \times \exp(-x^2) - 0.17576058. \]

Ex105. Over [−1, 1],
\[ f(x) = 0.02(12 + 3x - 3.5x^2 + 7.2x^3) \times [1 + \cos(4\pi x)] \times (1 + 0.8 \times \sin(3\pi x)) + 
0.0679996; \]

Ex106. Over [0.5, 100],
\[ x^{-12} - 2x^{-6} + 1.0 \]

Ex107. Over \([-10, 10]\),

\[
f(x) = - \sum_{i=1}^{5} i \sin(i + (i + 1)x) + 12.0312494
\]

Ex108. Over \([-2, 2]\),

\[
f(x) = 1 + \sin(\pi x)
\]

Ex109. Over \([-4.52, 4.52]\),

\[
f(x) = 2.0 + \sin(x^2) + \sin(2x^2/3) - 0.0940388;
\]

Ex201. Over \([-1, 1.5] \times [-1, 1.5]\),

\[
f(x_1, x_2) = [x_1^2 - 0.3\cos(3\pi x_1)] + [2x_2^2 - 0.4\cos(4\pi x_2)] + 0.7
\]

Ex202. Over \([-5, 5] \times [-5, 5]\),

\[
f(x_1, x_2) = 1.03163 + [4x_1^2 - 2.1x_1^4 + x_1^6/3] + x_1x_2 - 4[(1 - x_2^2)x_2^2]
\]

Ex203. Over \([-2, 2] \times [-2, 2]\),

\[
f(x_1, x_2) = [-2 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + (6x_1 + 3)x_2^2)][30
\]
\[
+ (2x_1 - 3x_2)^2(18 - 32x_1 + 39x_2^2 + (48 - 36x_1)x_2)] + 488225.321093
\]

Ex204: Beale function over \([-4.5, 4.5] \times [-4.5, 4.5]\),

\[
f(x_1, x_2) = [1.5 - x_1 + x_1 x_2]^2 + [2.25 - x_1 + x_1 x_2^2]^2 + [2.625 - x_1 + x_1 x_2^3]^2.
\]

Ex205. Over \([-1, 1.5] \times [-1, 1.5]\),

\[
f(x_1, x_2) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1)\cos(4\pi x_2) + 0.3
\]

Ex206. Over \([-1, 1.5] \times [-1, 1.5]\),

\[
f(x_1, x_2) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1 + 4\pi x_2) + 0.3
\]

Ex207. Over \([-5, 5] \times [-5, 5]\),

\[
f(x_1, x_2) = 1.5 + 4x_2^2 + \left(\frac{x_1^2}{3} - 2.1\right)x_1^4 + x_1x_2 - 4(1 - x_2^2)x_2^2 + 11.6246
\]

105
Ex208. Over \([-1, 1.5] \times [-1, 1.5],\)

\[ f(x_1, x_2) = [x_1^2 - \cos(18x_1)] + [x_2^2 - \cos(18x_2)] + 2; \]

Ex209. Over \([-100, 100] \times [-100, 100],\)

\[ f(x_1, x_2) = \left( \frac{1}{200} \right) \sum_{i=1}^{2} x_i^2 - \prod_{i=1}^{2} \cos(x_i/i^2) + 1 \]

Ex210. Branin function, over \([-5, 10] \times [0, 15],\)

\[ f(x_1, x_2) = \left( x_2 - \left( \frac{1.275}{\pi^2} \right) x_1^2 + \left( \frac{5}{\pi} \right) x_1 - 6.0 \right)^2 + 10(1-1/(8\pi))\cos(x_1) + 10 - 0.398 \]

Ex211. Over \([-2, 2] \times [1, 2],\)

\[ f(x_1, x_2) = x_1 \sin(1/x_1) + x_2 |x_1|, \text{ if } x_1 \neq 0 \]

\[ = 0, \text{ e/w} \]

Ex212. Shubert function in 2-d, over \([-10, 10] \times [-10, 10],\)

\[ f(x_1, x_2) = \sum_{i=1}^{5} i \cos((i+1)x_1 + i) \times \sum_{i=1}^{5} i \cos((i+1)x_2 + i) + 186.7309 \]

Ex213. Modified Shubert function, over \([-10, 10] \times [-10, 10],\)

\[ f(x_1, x_2) = \sum_{i=1}^{5} i \cos((i+1)x_1 + i) \times \sum_{i=1}^{5} i \cos((i+1)x_2 + i) + 186.73091 \]

\[ + (x_1 + 1.42513)^2 + (x_2 + 0.80032)^2 \]

Ex301. Bunnag's 302 De Jong Function F1: over \([-5.12, 5.12]^3,\)

\[ f(x) = x_1^2 + x_2^2 + x_3^2 \]

Ex302. Over \([-1, 2]^3,\)

\[ f(x) = x_1^2 + 4x_2^2 + 6x_3^2 + 14x_1x_2 + 6x_1x_3 - 10x_2x_3 + 19 \]
Ex303. Over \([-1, 2]^3\),
\[ f(x) = x_1^2 + x_2^2 + x_3^2 \]

Ex304. Over \([-1, 2]^3\),
\[ f(x) = 2x_1^2 + 2x_2^2 + 2x_3^2 + 2x_1x_2 + 2x_1x_3 + 2x_2x_3 \]

Ex305. Over \([-1, 2]^3\),
\[ f(x) = x_1^2 + x_2^2 + x_3^2 + 4x_1x_2 + 6x_1x_3 + 4x_2x_3 + 9.48163 \]

Ex306. Over \([-1, 2]^3\),
\[ f(x) = x_1^2 + x_2^2 + x_3^2 + 8x_1x_2 + 10x_1x_3 + 8x_2x_3 + 20.6091 \]

Ex401. Over \([0.5, 1] \times [0.5, 2]^3\),
\[ f(x) = 4(x_1 - 1)^2 - (x_2 - 1)^4 - 2(x_3 - 2)^2 + (x_4 - 2)^4 + 5.5 \]

Ex402. Over \([0.2, 0.4] \times [0.2, 0.8]^3\),
\[ f(x) = 4(x_1 - 1)^2 - (x_2 - 1)^4 - 2(x_3 - 2)^2 + (x_4 - 2)^4 + 0.1(x_1 - 1)(x_3 - 2) \]
\[ + 0.1(x_4 - 2)(x_1 - 1) + 3.196 \]

Ex403. Over \([0.5, 1] \times [0.5, 2]^3\),
\[ f(x) = 4(x_1 - 1)^2 - (x_2 - 1)^4 - 2(x_3 - 2)^2 + (x_4 - 2)^4 + 0.1(x_1 - 1)(x_3 - 2) \]
\[ + 0.1(x_4 - 2)(x_1 - 1) + 0.1(x_1 - 1)(x_2 - 1) + 0.1(x_2 - 1)(x_3 - 2) \]
\[ + 0.1(x_2 - 1)(x_4 - 2) + 0.1(x_3 - 2)(x_4 - 2) + 5.65 \]

Ex404. Over \([0.5, 1] \times [0.5, 2]^3\),
\[ f(x) = 4(x_1 - 1)^2 - (x_2 - 1)^4 - 2(x_3 - 2)^2 + (x_4 - 2)^4 + 2(x_1 - 1)(x_3 - 2) \]
\[ + 2(x_4 - 2)(x_1 - 1) + 5.5 \]

Ex405. Over \([0.5, 1] \times [0.5, 2]^3\),
\[ f(x) = 4(x_1 - 1)^2 - (x_2 - 1)^4 - 2(x_3 - 2)^2 + (x_4 - 2)^4 + 2(x_1 - 1)(x_2 - 1) \\
+ 2(x_1 - 1)(x_3 - 2) + 2(x_4 - 2)(x_1 - 1) + 4(x_2 - 1)(x_3 - 2) \\
+ 4(x_2 - 1)(x_4 - 2) + 4(x_3 - 2)(x_4 - 2) + 11.5 \]

Ex406. Over \([0.2, 0.4] \times [0.2, 0.8]^3\),
\[ f(x) = 4(x_1 - 1)^2 - (x_2 - 1)^4 - 2(x_3 - 2)^2 + (x_4 - 2)^4 + 20 \left( x_1 + \frac{1}{3} \right) \left( x_3 - \frac{2}{3} \right) + 9.70051 \]

Ex407. Over \([0.2, 0.4] \times [0.2, 0.8]^3\),
\[ f(x) = 4(x_1 - 1)^2 - (x_2 - 1)^4 - 2(x_3 - 2)^2 + (x_4 - 2)^4 + 20(x_1 - 1)(x_2 - 1) \\
+ 20(x_1 - 1)(x_3 - 2) + 20(x_1 - 1)(x_4 - 2) + 40(x_2 - 1)(x_3 - 2) \\
+ 40(x_2 - 1)(x_4 - 2) + 40(x_3 - 2)(x_4 - 2) - 108.6334732 \]

Ex701. Over \([1, 2] \times [1, 5]^6\),
\[ f(x) = 4(x_1 - 1)^2 - (x_2 - 2)^4 - 2(x_3 - 4)^2 + (x_4 - 3)^4 + (x_5 - 2) - (x_6 - 3)^2 + |x_7 - 4| \]

Ex702. Over \([0.5, 1.5] \times [0.5, 2.5]^6\),
\[ f(x) = 4(x_1 - 1)^2 - (x_2 - 2)^4 - 2(x_3 - 4)^2 + (x_4 - 3)^4 + (x_5 - 2) - (x_6 - 3)^2 + |x_7 - 4| \\
+ 20 \left( x_1 + \frac{1}{3} \right) \left( x_3 - \frac{4}{3} \right) + 57.9719 \]

Ex703. Over \([0.5, 1.5] \times [0.5, 2.5]^6\),
\[ f(x) = 4(x_1 - 1)^2 - (x_2 - 2)^4 - 2(x_3 - 4)^2 + (x_4 - 3)^4 + (x_5 - 2) - (x_6 - 3)^2 + |x_7 - 4| \\
+ 20(x_1 - 1)(x_2 - 2) + 20(x_1 - 1)(x_3 - 4) + 20(x_1 - 1)(x_4 - 3) \\
+ 40(x_2 - 2)(x_4 - 3) + 40(x_3 - 4)(x_4 - 3) + 20(x_5 - 2)(x_6 - 3) \\
- 40(x_6 - 3)(x_7 - 4) + 411.747074 \]

Ex9901. Rosenbrock's function over \([-100, 100]^n\),
\[ f(x) = \sum_{i=1}^{n-1}[100(x_{i+1} - x_i^2)^2 + (1-x_i)^2]. \]

If \( n = 4 \)

s.t. \( x_1 + x_2 + x_3 + x_4 = 4; \)

\( x_1 - x_2 + x_3 - x_4 = 0. \)

If \( n = 10 \)

s.t. \( x_1 + x_2 + x_3 + x_4 + \ldots + x_{39} + x_{40} = 10; \)

\( x_1 - x_2 + x_3 - x_4 + \ldots + x_{39} - x_{40} = 0. \)

If \( n = 40 \)

s.t. \( x_1 + x_2 + x_3 + x_4 + \ldots + x_{99} + x_{100} = 100; \)

\( x_1 - x_2 + x_3 - x_4 + \ldots + x_{99} - x_{100} = 0. \)

If \( n = 100 \)

s.t. \( x_1 + x_2 + x_3 + x_4 + \ldots + x_{999} + x_{1000} = 100; \)

\( x_1 - x_2 + x_3 - x_4 + \ldots + x_{999} - x_{1000} = 0. \)

Ex9902. Zakharov's function in any dim, over \([-9, 11]^n\),

\[ f(x) = \sum_{i=1}^{n} x_i^2 + 0.5 \sum_{i=1}^{n} i x_i \]2 + \( 0.5 \sum_{i=1}^{n} i x_i \)4

If \( n = 4 \)

s.t. \( x_1 - 2x_2 + 3x_3 + x_4 = 0; \)

\( x_1 + x_2 + x_3 + x_4 = 0. \)

If \( n = 10 \)

s.t. \( x_1 - 2x_2 + 3x_3 + x_4 - 2x_5 + 3x_6 + x_7 - 2x_8 + 3x_9 + x_{10} = 0; \)

\( x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} = 0. \)

If \( n = 40 \)
s.t. \( x_1 - 2x_2 + 3x_3 + x_4 - 2x_5 + 3x_6 + \cdots + x_{37} - 2x_{38} + 3x_{39} + x_{40} = 0; \)
\[
x_1 + x_2 + x_3 + x_4 + \cdots + x_{39} + x_{40} = 0.
\]

If \( n = 100 \)

s.t. \( x_1 - 2x_2 + 3x_3 + x_4 + -2x_5 + 3x_6 + \cdots + x_{97} - 2x_{98} + 3x_{99} + x_{100} = 0; \)
\[
x_1 + x_2 + x_3 + x_4 + \cdots + x_{99} + x_{100} = 0.
\]

Ex9903. Sphere function in any dim over whole space, over \([-95,105]^{n}\),
\[
f(x) = \sum_{i=1}^{n} x_i^2
\]

If \( n = 4; \)

s.t. \( x_4 = 0; \)
\[
x_1 + x_2 + x_4 = 0.
\]

If \( n = 10; \)

s.t. \( x_4 + x_8 = 0; \)
\[
x_1 + x_2 + x_4 + x_5 + x_6 + x_8 + x_9 + x_{10} = 0.
\]

If \( n = 40; \)

s.t. \( x_4 + x_8 + \cdots + x_{28} + x_{32} + x_{36} + x_{40} = 0; \)
\[
x_1 + x_2 + x_4 + x_5 + x_6 + x_8 + \cdots + x_{37} + x_{38} + x_{40} = 0.
\]

If \( n = 100; \)

s.t. \( x_4 + x_8 + \cdots + x_{88} + x_{92} + x_{96} + x_{100} = 0; \)
\[
x_1 + x_2 + x_4 + x_5 + x_6 + x_8 + \cdots + x_{97} + x_{98} + x_{100} = 0.
\]

Ex9904. Schwefel's function 2.22 in any dim over \([-10,8]^{n}\),
\[
f(x) = \sum_{i=1}^{n} |x_i| + \prod_{i=1}^{n} |x_i|
\]

If \( n = 4 \)
s.t.  \( x_2 + x_3 + x_4 = 0; \)
\[ x_1 - x_2 + 2x_3 + x_4 = 0. \]

If \( n = 10 \)

s.t.  \( x_2 + x_3 + x_4 + x_6 + x_7 + x_8 + x_{10} = 0; \)
\[ x_1 - x_2 + 2x_3 + x_4 + x_5 - x_6 + 2x_7 + x_8 + x_9 - x_{10} = 0. \]

If \( n = 40 \)

s.t.  \( x_2 + x_3 + x_4 + x_6 + x_7 + x_8 + \cdots + x_{36} + x_{38} + x_{39} + x_{40} = 0; \)
\[ x_1 - x_2 + 2x_3 + x_4 + x_5 - x_6 + 2x_7 + x_8 \cdots + 2x_{39} + x_{40} = 0. \]

If \( n = 100 \)

s.t.  \( x_2 + x_3 + x_4 + x_6 + x_7 + x_8 + \cdots + x_{96} + x_{98} + x_{99} + x_{100} = 0; \)
\[ x_1 - x_2 + 2x_3 + x_4 + x_5 - x_6 + 2x_7 + x_8 \cdots + 2x_{99} + x_{100} = 0. \]

Similarly, the following problem’s constraints are repeated in the same way

Ex9905. Schwefel's function 2.21, over \([-100, 80]^n\),

\[ f(x) = \max\{|x_i|: i = 1, \ldots, n\}. \]

s.t.  \( x_1 - 2x_2 + 3x_3 + x_4 + \cdots = 0; \)
\[ x_1 + x_2 + x_3 + x_4 + \cdots = 0. \]

Ex9906. A step function in any dim, over \([-100,200]^n\),

\[ f(x) = \sum_{i=1}^n ([x_i + 0.5])^2, \] where \([.]\) = the greatest integer function.

s.t.  \( x_1 + x_2 + x_3 + x_4 + \cdots = 0; \)
\[ x_1 - 2x_2 + 0x_3 + 0x_4 + \cdots = 0. \]

Ex9907. Generalized Rastrigin's function over \([-6.12,5.12]^n\) in any dim \( n \),

\[ f(x) = \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i) + 10] \]
s.t. $x_1 + x_2 + x_3 + \cdots = 0$;
\[ x_1 - 2x_2 + 0x_3 + \cdots = 0. \]

Ex9908. Ackley's function in any dim, over $[-31,33]^n$,
\[
f(x) = -20 \exp\{-0.2[\sum_{i=1}^{n} x_i^2/n]^{1/2}\} - \exp\{\sum_{i=1}^{n} \cos(2\pi x_i)/n\} + 22.7182818
\]
s.t. $x_1 + x_2 + x_3 + x_4 + \cdots = 0$;
\[ x_1 + 0x_2 + 2x_3 + 3x_4 + \cdots = 0. \]

Ex9909. Modified Griewank's function over $[-590, 600]^n$ in any dim $n$,
\[
f(x) = 1 + \sum_{i=1}^{n} x_i^2/4000 - \prod_{i=1}^{n} \cos(x_i/\sqrt{i})
\]
s.t. $x_1 + x_2 + x_3 + x_4 + \cdots = 0$;
\[ x_1 - 2x_2 + 0x_3 + 3x_4 + \cdots = 0. \]

Ex9910. M. Locatelli's modification #2 of Griewank's function in any dim, over $[-590,600]^n$,
\[
f(x) = \sum_{i=1}^{n} x_i^2/4000 - \sum_{i=1}^{n} \ln[2 + \cos(x_i/\sqrt{i})] + n \ln 3.
\]
s.t. $x_1 + x_2 + x_3 + \cdots = 0$;
\[ x_1 - 2x_2 + 5x_3 + \cdots = 0. \]

Ex9911. M. Locatelli's modification #3 of Griewank's function in any dim, over $[-590,600]^n$,
\[
f(x) = \sum_{i=1}^{n} x_i^2/4000 - \sum_{i=1}^{n} \ln[2 + \cos(\sum_{j=1}^{n} A_{ij} x_j)] + n \ln 3.
\]
s.t. $x_1 + x_2 + x_3 + \cdots = 0$;
\[ 0x_1 + 5x_2 + 6x_3 + \cdots = 0. \]
where $A_{ij} = 1$ if $i \neq j$, and $A_{ii} = n + 1$.

Ex9912. Over $[-590,600]^n$,
\[
f(x) = \{n - \sum_{j=1}^{n} [\sin(x_j) + \cos(x_j)]\}^2;
\]
s.t. $x_1 + 2x_2 + x_3 - x_4 + \cdots = 0$;

$0.3x_1 + 0.1x_2 + 3x_3 + 9x_4 + \cdots = 0$.

Ex9913. A Powell’s over $[-590,600]^n$,

$$f(x) = |n - \sum_{j=1}^n [\sin(x_j) + \cos(x_j)]|;$$

s.t. $x_1 + x_2 - x_3 - x_4 + \cdots = 0$;

$0.03x_1 + 0.01x_2 - 9x_3 + 9x_4 + \cdots = 0$.

Ex9914. Modified Zakharov's function in any dim, Over $[-10^{1/2}, 10^{1/2}]^4 \times [-10, 10]^{n-4}$,

$$f(x) = \sum_{i=1}^n x_i^2 + \left(0.5 \sum_{i=1}^n ix_i\right)^2 + \left(0.5 \sum_{i=1}^n ix_i\right)^4;$$

If $n = 4$

s.t. $x_1 + 3x_2 - 2x_3 = 2$;

$x_1 + x_4 = 2$.

If $n = 10$

s.t. $x_1 + 3x_2 - 2x_3 + x_5 + 3x_6 - 2x_7 + x_9 + 3x_{10} = 8$;

$x_1 + x_4 + x_5 + x_8 + x_9 = 5$.

If $n = 40$

s.t. $x_1 + 3x_2 - 2x_3 + x_5 + 3x_6 - 2x_7 + \cdots + x_{37} + 3x_{38} - 2x_{39} = 20$;

$x_1 + x_4 + x_5 + x_8 + \cdots + x_{39} + x_{40} = 20$.

If $n = 100$

s.t. $x_1 + 3x_2 - 2x_3 + x_5 + 3x_6 - 2x_7 + \cdots + x_{97} + 3x_{98} - 2x_{99} = 50$;

$x_1 + x_4 + x_5 + x_8 + \cdots + x_{99} + x_{100} = 50$.

Ex9915. A modified sphere function in any dim, over $[-95^{1/2}, 105^{1/2}]^n \times [-95,105]^{n-4}$,

$$f(x) = \sum_{i=1}^n x_i^2$$
If \( n = 4 \)
\[
\text{s.t. } x_1 + x_2 + 3x_3 - 2x_4 = 3; \\
-x_1 + 4x_2 + x_3 + 3x_4 = 7.
\]
If \( n = 10 \)
\[
\text{s.t. } x_1 + x_2 + 3x_3 - 2x_4 + x_5 + x_6 + 3x_7 - 2x_8 + x_9 + x_{10} = 8; \\
-x_1 + 4x_2 + x_3 + 3x_4 - x_5 + 4x_6 + x_7 + 3x_8 - x_9 + 4x_{10} = 17.
\]
If \( n = 40 \)
\[
\text{s.t. } x_1 + x_2 + 3x_3 - 2x_4 + x_5 + x_6 + 3x_7 - 2x_8 + \cdots + x_{37} + x_{38} + 3x_{39} - 2x_{40} = 30; \\
-x_1 + 4x_2 + x_3 + 3x_4 - x_5 + 4x_6 + x_7 + 3x_8 + \cdots + x_{39} + 3x_{40} = 28.
\]
If \( n = 100 \)
\[
\text{s.t. } x_1 + x_2 + 3x_3 - 2x_4 + x_5 + x_6 + 3x_7 - 2x_8 + \cdots + x_{97} + x_{98} + 3x_{99} - 2x_{100} = 75; \\
-x_1 + 4x_2 + x_3 + 3x_4 - x_5 + 4x_6 + x_7 + 3x_8 + \cdots + x_{99} + 3x_{100} = 175.
\]
Ex9916. Schwefel's function 2.22 in any dim, over \([-10^{1/2}, 10^{1/2}]^4 \times [-10, 10]^{n-4}\),
\[
f(x) = \sum_{i=1}^{n} |x_i| + \prod_{i=1}^{n} |x_i|
\]
If \( n = 4 \)
\[
\text{s.t. } -0.57735x_1 + x_2 - 2x_3 + x_4 = -0.57735; \\
-3x_1 + 0.5x_2 + 0x_3 + x_4 = -1.5.
\]
If \( n = 10 \)
\[
\text{s.t. } -0.57735x_1 + x_2 - 2x_3 + x_4 - 0.57735x_5 + x_6 - 2x_7 + x_8 - 0.57735x_9 + x_{10} = -0.73205; \\
-3x_1 + 0.5x_2 + 0x_3 + x_4 - 3x_5 + 0.5x_6 + 0x_7 + x_8 - 3x_9 + 0.5x_{10} = -5.5.
\]
If \( n = 40 \)
s.t. \(-0.57735x_1 + x_2 - 2x_3 + x_4 - 0.57735x_5 + x_6 - 2x_7 + x_8 + \cdots - 0.57735x_{37} + x_{38} - 2x_{39} + x_{40} = -5.7735;
\)

\[-3x_1 + 0.5x_2 + 0x_3 + x_4 - 3x_5 + 0.5x_6 + 0x_7 + x_8 + \cdots + 0x_{39} + x_{40} = -15.\]

If \(n = 100\)

s.t. \(-0.57735x_1 + x_2 - 2x_3 + x_4 - 0.57735x_5 + x_6 - 2x_7 + x_8 + \cdots - 0.57735x_{97} + x_{98} - 2x_{99} + x_{100} = -14.43375\)

\[-3x_1 + 0.5x_2 + 0x_3 + x_4 - 3x_5 + 0.5x_6 + 0x_7 + x_8 + \cdots + 0x_{99} + x_{100} = -37.5.\]

Ex9917. Schwefel's function 1.2 in any dim, over \([-10^{1/2}, 10^{1/2}]^n \times [-10, 10]^{n-4},\]

\[f(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2\]

If \(n = 4\)

s.t. \(x_1 + x_2 - x_3 + x_4 = 2;\)

\(-x_1 + x_2 - x_3 - x_4 = -2.\)

If \(n = 10\)

s.t. \(x_1 + x_2 - x_3 + x_4 + x_5 - x_6 + x_7 + x_8 - x_9 + x_{10} = 4;\)

\(-x_1 + x_2 - x_3 - x_4 + x_5 - x_6 - x_7 + x_8 - x_9 - x_{10} = -4.\)

If \(n = 40\)

s.t. \(x_1 + x_2 - x_3 + x_4 + x_5 - x_6 + \cdots + x_{37} + x_{38} - x_{39} + x_{40} = 14\)

\(-x_1 + x_2 - x_3 - x_4 + x_5 - x_6 + \cdots - x_{39} - x_{40} = -14.\)

If \(n = 100\)

s.t. \(x_1 + x_2 - x_3 + x_4 + x_5 - x_6 + \cdots + x_{97} + x_{98} - x_{99} + x_{100} = 34\)

\(-x_1 + x_2 - x_3 - x_4 + x_5 - x_6 + \cdots - x_{99} - x_{100} = -34.\)
Ex9918. Schwefel's function 2.21 over \([-100^{1/2}, 100^{1/2}]^4 \times [-100, 100]^{n-4}\) in any dim \(n\),

\[f(x) = \max\{|x_i|: i = 1, \ldots, n\}.
\]

If \(n = 4\)

s.t. \(12x_1 + 5x_2 + x_3 - 12x_4 = 6;\)

\[12x_1 + 7x_2 + 0x_3 - 18x_4 = 1.
\]

If \(n = 10\)

s.t. \(12x_1 + 5x_2 + x_3 - 12x_4 + 12x_5 + 5x_6 + x_7 - 12x_8 + 12x_9 + 5x_{10} = 29;\)

\[12x_1 + 7x_2 + 0x_3 - 18x_4 + 12x_5 + 7x_6 + 0x_7 - 18x_8 + 12x_9 + 7x_{10} = 21.
\]

If \(n = 40\)

s.t. \(12x_1 + 5x_2 + x_3 - 12x_4 + 12x_5 + 5x_6 + x_7 - 12x_8 + \cdots + 12x_{37} + 5x_{38} + x_{39} - 12x_{40} = 60;\)

\[12x_1 + 7x_2 + 0x_3 - 18x_4 + 12x_5 + 7x_6 + 0x_7 - 18x_8 + \cdots + 0x_{39} - 18x_{40} = 10.
\]

If \(n = 100\)

s.t. \(12x_1 + 5x_2 + x_3 - 12x_4 + 12x_5 + 5x_6 + x_7 - 12x_8 + \cdots + 12x_{97} + 5x_{98} + x_{99} - 12x_{100} = 150;\)

\[12x_1 + 7x_2 + 0x_3 - 18x_4 + 12x_5 + 7x_6 + 0x_7 - 18x_8 + \cdots + 0x_{99} - 18x_{100} = 25.
\]

Ex9919. Over \([-10^{1/2}, 10^{1/2}]^4 \times [-10, 10]^{n-4}\) in any dim \(n > 1\),

\[f(x) = \sum_{i=1}^{n-1}[100(x_{i+1}^2 - x_i^4)^2 + (1 - x_i^2)^2].
\]

If \(n = 4\)

s.t. \(x_1 + 0.7x_2 - 2x_3 + x_4 = 0.7;\)

\[0x_1 + x_2 + 2.1x_3 - 0x_4 = 3.1.
\]

If \(n = 10\)

s.t. \(x_1 + 0.7x_2 - 2x_3 + x_4 + 0.7x_5 - 2x_6 + x_7 + 0.7x_8 - 2x_9 + x_{10} = 0.1;\)

\[g(x) = \sum_{i=1}^{n-1}[100(x_{i+1}^2 - x_i^4)^2 + (1 - x_i^2)^2].
\]
\[0x_1 + x_2 + 2.1x_3 - 0x_4 + x_5 + 2.1x_6 - 0x_7 + x_8 + 2.1x_9 - 0x_{10} = 9.3.\]

If \( n = 40 \)
\[
s.t. \ x_1 + 0.7x_2 - 2x_3 + x_4 + 0.7x_5 - 2x_6 + \cdots + x_{37} + 0.7x_{38} - 2x_{39} + x_{40} = -2.9
\]
\[0x_1 + x_2 + 2.1x_3 - 0x_4 + x_5 + 2.1x_6 + \cdots + 2.1x_{39} - 0x_{40} = 40.3.\]

If \( n = 100 \)
\[
s.t. \ x_1 + 0.7x_2 - 2x_3 + x_4 + 0.7x_5 - 2x_6 + \cdots + x_{97} + 0.7x_{98} - 2x_{99} + x_{100} = -8.9
\]
\[0x_1 + x_2 + 2.1x_3 - 0x_4 + x_5 + 2.1x_6 + \cdots + 2.1x_{99} - 0x_{100} = 102.3.\]

Ex9920. Generalized Rastrigin's function over \([-6.12,6.12] 4 \]
\[f(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x) + 10] \text{ [exact range each term]}\]

If \( n = 4 \)
\[
s.t. \ x_1 - 3x_2 - 2x_3 + 2x_4 = -2;
\]
\[x_1 + x_2 + 2x_3 - 3x_4 = 1.\]

If \( n = 10 \)
\[
s.t. \ x_1 - 3x_2 - 2x_3 + 2x_4 + x_5 - 3x_6 - 2x_7 + 2x_8 + x_9 - 3x_{10} = -6;
\]
\[x_1 + x_2 + 2x_3 - 3x_4 + x_5 + x_6 + 2x_7 - 3x_8 + x_9 + x_{10} = 4.\]

If \( n = 40 \)
\[
s.t. \ x_1 - 3x_2 - 2x_3 + 2x_4 + x_5 - 3x_6 - 2x_7 + 2x_8 + \cdots + x_{37} - 3x_{38} - 2x_{39} + 2x_{40} = -20;
\]
\[x_1 + x_2 + 2x_3 - 3x_4 + x_5 + x_6 + 2x_7 - 3x_8 + \cdots + 2x_{39} - 3x_{40} = 10.\]

If \( n = 100 \)
\[
s.t. \ x_1 - 3x_2 - 2x_3 + 2x_4 + x_5 - 3x_6 - 2x_7 + 2x_8 + \cdots + x_{97} - 3x_{98} - 2x_{99} + 2x_{100} = -50;
\]
\[x_1 + x_2 + 2x_3 - 3x_4 + x_5 + x_6 + 2x_7 - 3x_8 + \cdots + 2x_{99} - 3x_{100} = 25.\]
Ex9921. Ackley's function in any dim over \([-32, 32]^n\),

$$f(x) = -20 \exp\{-0.2[\sum_{i=1}^{n} x_i^2 / n]^{1/2}\} - \exp\{\sum_{i=1}^{n} \cos(2\pi x_i) / n\} + 22.7182818$$

If \(n = 4\)

s.t. \(6x_1 + 3x_2 + 3x_3 + 2x_4 = 14;\)

\(10x_1 + 0x_2 + 10x_3 + x_4 = 21.\)

If \(n = 10\)

s.t. \(6x_1 + 3x_2 + 3x_3 + 2x_4 + 6x_5 + 3x_6 + 3x_7 + 2x_8 + 6x_9 + 3x_{10} = 37;\)

\(10x_1 + 0x_2 + 10x_3 + x_4 + 10x_5 + 0x_6 + 10x_7 + x_8 + 10x_9 + 0x_{10} = 52.\)

If \(n = 40\)

s.t. \(6x_1 + 3x_2 + 3x_3 + 2x_4 + 6x_5 + 3x_6 + 3x_7 + 2x_8 + \cdots + 6x_{37} + 3x_{38} + 3x_{39} + 2x_{40} = 140;\)

\(10x_1 + 0x_2 + 10x_3 + x_4 + 10x_5 + 0x_6 + 10x_7 + x_8 + \cdots + 10x_{39} + x_{40} = 210.\)

If \(n = 100\)

s.t. \(6x_1 + 3x_2 + 3x_3 + 2x_4 + 6x_5 + 3x_6 + 3x_7 + 2x_8 + \cdots + 6x_{97} + 3x_{98} + 3x_{99} + 2x_{100} = 350;\)

\(10x_1 + 0x_2 + 10x_3 + x_4 + 10x_5 + 0x_6 + 10x_7 + x_8 + \cdots + 10x_{99} + x_{100} = 525.\)

Ex9922. Modified Griewank's function over \([-600, 600]^n\) in any dim \(n\),

$$f(x) = 1 + \sum_{i=1}^{n} x_i^2 / 4000 - \prod_{i=1}^{n} \cos(x_i / \sqrt{i})$$

If \(n = 4\)

s.t. \(x_1 + 2x_2 + 8x_3 + x_4 = 12;\)

\(3x_1 + 5x_2 + 0x_3 + 0x_4 = 8.\)

If \(n = 10;\)

\(\)
s.t. $x_1 + 2x_2 + 8x_3 + x_4 + x_5 + 2x_6 + 8x_7 + x_8 + x_9 + 2x_{10} = 27$;

$3x_1 + 5x_2 + 0x_3 + 0x_4 + 3x_5 + 5x_6 + 0x_7 + 0x_8 + 3x_9 + 5x_{10} = 24$.

If $n = 40$;

s.t. $x_1 + 2x_2 + 8x_3 + x_4 + x_5 + 2x_6 + 8x_7 + x_8 + \ldots + x_{37} + 2x_{38} + 8x_{39} + x_{40} = 120$;

$3x_1 + 5x_2 + 0x_3 + 0x_4 + 3x_5 + 5x_6 + 0x_7 + 0x_8 + \ldots + 0x_{39} + 0x_{40} = 80$.

If $n = 100$;

s.t. $x_1 + 2x_2 + 8x_3 + x_4 + x_5 + 2x_6 + 8x_7 + x_8 + \ldots + x_{97} + 2x_{98} + 8x_{99} + x_{100} = 300$;

$3x_1 + 5x_2 + 0x_3 + 0x_4 + 3x_5 + 5x_6 + 0x_7 + 0x_8 + \ldots + 0x_{99} + 0x_{100} = 200$.

Ex9923. Locatelli's modification #1 of Griewank's function in any dim over $[-600, 600]^n$,

$$f(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n}[2 + \cos(x_i/\sqrt{i})]/3 + 1.$$

If $n = 4$

s.t. $-x_1 - 0.5x_2 - 0.25x_3 - 0.25x_4 = -2$;

$x_1 - 0.25x_2 - 1.5x_3 + x_4 = 0.25$.

If $n = 10$

s.t. $-x_1 - 0.5x_2 - 0.25x_3 - 0.25x_4 - x_5 - 0.5x_6 - 0.25x_7 - 0.25x_8 - x_9 - 0.5x_{10} = -5.5$;

$x_1 - 0.25x_2 - 1.5x_3 + x_4 + x_5 - 0.25x_6 - 1.5x_7 + x_8 + x_9 - 0.25x_{10} = 1.25$.

If $n = 40$

s.t. $-x_1 - 0.5x_2 - 0.25x_3 - 0.25x_4 - x_5 - 0.5x_6 - 0.25x_7 - 0.25x_8 + \ldots + x_{37} - 0.5x_{38} - 0.25x_{39} - 0.25x_{40} = -20$;

$x_1 - 0.25x_2 - 1.5x_3 + x_4 + x_5 - 0.25x_6 - 1.5x_7 + x_8 + \ldots - 1.5x_{39} + x_{40} = 2.5$.

If $n = 100$
\[
s.t. \quad -x_1 - 0.5x_2 - 0.25x_3 - 0.25x_4 - x_5 - 0.5x_6 - 0.25x_7 - 0.25x_8 + \cdots - x_{97} - 0.5x_{98} - 0.25x_{99} - 0.25x_{100} = -50; \\
x_1 - 0.25x_2 - 1.5x_3 + x_4 + x_5 - 0.25x_6 - 1.5x_7 + x_8 + \cdots - 1.5x_{99} + x_{100} = 6.25.
\]

Ex9924. Locatelli's modification #2 of Griewank's function in any dim, over \([-600,600]^n\),

\[
f(x) = \sum_{i=1}^n x_i^2 / 4000 - \sum_{i=1}^n \ln[2 + \cos(x_i / \sqrt{i})] + n \ln 3.
\]

If \(n = 4\)

s.t. \(0.2x_1 + 2x_2 + 0.1x_3 - 4x_4 = -1.7;\)
\(-4x_1 - x_2 + 5x_3 + 3x_4 = 3.\)

If \(n = 10\)

s.t. \(0.2x_1 + 2x_2 + 0.1x_3 - 4x_4 + 0.2x_5 + 2x_6 + 0.1x_7 - 4x_8 + 0.2x_9 + 2x_{10} = -1.2;\)
\(-4x_1 - x_2 + 5x_3 + 3x_4 - 4x_5 - x_6 + 5x_7 + 3x_8 - 4x_9 - x_{10} = 1.\)

If \(n = 40\)

s.t. \(0.2x_1 + 2x_2 + 0.1x_3 - 4x_4 + 0.2x_5 + 2x_6 + 0.1x_7 - 4x_8 + \cdots + 0.2x_{37} + 2x_{38} + 0.1x_{39} - 4x_{40} = -17;\)
\(-4x_1 - x_2 + 5x_3 + 3x_4 - 4x_5 - x_6 + 5x_7 + 3x_8 + \cdots + 5x_{39} + 3x_{40} = 30.\)

If \(n = 100\)

s.t. \(0.2x_1 + 2x_2 + 0.1x_3 - 4x_4 + 0.2x_5 + 2x_6 + 0.1x_7 - 4x_8 + \cdots + 0.2x_{97} + 2x_{98} + 0.1x_{99} - 4x_{100} = -42.5;\)
\(-4x_1 - x_2 + 5x_3 + 3x_4 - 4x_5 - x_6 + 5x_7 + 3x_8 + \cdots + 5x_{99} + 3x_{100} = 75.\)

Ex9925. Locatelli's modification #3 of Griewank's function in any dim, over \([-600,600]^n\),

\[
f(x) = \sum_{i=1}^n x_i^2 / 4000 - \sum_{i=1}^n \ln[2 + \cos(\sum_{j=1}^n A_{ij}x_j)] + n \ln 3.
\]

If \(n = 4\)
s.t.  
\[-x_1 - 0.5x_2 + 2x_3 + x_4 = 1.5;\]
\[-x_1 + 2x_2 + x_3 - 2x_4 = 0.\]

If \( n = 10 \)

\[-x_1 - 0.5x_2 + 2x_3 + x_4 - x_5 - 0.5x_6 + 2x_7 + x_8 - x_9 - 0.5x_{10} = 1.5;\]
\[-x_1 + 2x_2 + x_3 - 2x_4 - x_5 + 2x_6 + x_7 - 2x_8 - x_9 + 2x_{10} = 1.\]

If \( n = 40 \)

\[-x_1 - 0.5x_2 + 2x_3 + x_4 - x_5 - 0.5x_6 + 2x_7 + x_8 + \cdots - x_{37} - 0.5x_{38} + 2x_{39} + x_{40} = 15;\]
\[-x_1 + 2x_2 + x_3 - 2x_4 - x_5 + 2x_6 + x_7 - 2x_8 + \cdots + x_{39} - 2x_{40} = 0.\]

If \( n = 100 \)

\[-x_1 - 0.5x_2 + 2x_3 + x_4 - x_5 - 0.5x_6 + 2x_7 + x_8 + \cdots - x_{97} - 0.5x_{98} + 2x_{99} + x_{100} = 37.5;\]
\[-x_1 + 2x_2 + x_3 - 2x_4 - x_5 + 2x_6 + x_7 - 2x_8 + \cdots + x_{99} - 2x_{100} = 0.\]