Journal of Computational Mathematics, Vol.22, No.6, 2004, 895–904.

A BRANCH AND BOUND ALGORITHM FOR SEPARABLE CONCAVE PROGRAMMING *1)

Hong-gang Xue
(*Department of Mathematics, Faculty of Science, Xi'an Jiaotong University, Xi'an 710049, China*)
(*Department of Mathematics, Faculty of Science, Xi'an University of Technology, Xi'an 710049, China*)

Cheng-xian Xu   Feng-min Xu
(*Department of Mathematics, Faculty of Science, Xi'an Jiaotong University, xi'an 710049, China*)

Abstract

In this paper, we propose a new branch and bound algorithm for the solution of large scale separable concave programming problems. The largest distance bisection (**LDB**) technique is proposed to divide rectangle into sub-rectangles when one problem is branched into two subproblems. It is proved that the **LDB** method is a normal rectangle subdivision(**NRS**). Numerical tests on problems with dimensions from 100 to 10000 show that the proposed branch and bound algorithm is efficient for solving large scale separable concave programming problems, and convergence rate is faster than ω-subdivision method.

Mathematics subject classification: 49M37, 65K05, 90C30.
Key words: Branch and bound algorithm, Separable programming, Largest distance bisection, Normal rectangle subdivision, ω-subdivision.

1. Introduction

The nonconvex programming has come a long way from 60th years of last century (see[6], [9], [4]). It is an important area of mathematical programming, and has many applications in economics, finance, planning, and engineering design. The separable concave programming is a kind of special problems in nonconvex programming. This paper studies the solution of the separable concave programming over a polytope in the form

$$\max\{f(x)|x \in V,\ l \le x \le u\}, \qquad (\mathbf{SCP})$$

where the objective function has the format

$$f(x) = \sum_{i=1}^{n} f_i(x_i), \qquad (1.1)$$

$f_i(x_i), i = 1, 2, \ldots, n$ are convex functions, $x = (x_1, x_2, \ldots, x_n)^T$. The feasible region is the intersection of the polytope V and the rectangle with low bounds $l = (l_1, l_2, \ldots, l_n)^T$ and upper bounds $u = (u_1, u_2, \ldots, u_n)^T$.

Algorithms have been proposed to solve problems (SCP). Falk and Soland(1969) reported a branch and bound algorithm in [1], Thai Quynh Phong et al(1995) proposed a decomposition branch and bound method in [8] to find the global solution of indefinite quadratic programming problems, and tests on problems with 20 concave variables and 200 convex variables show the efficiency of the method. Konno (2001) proposed a branch and bound algorithm to solve large

* Received May 13, 2003; final revised December 11, 2003.
1) This work is supported by National Natural Key Product Foundations of China 10231060.

scale portfolio optimization problems with concave transaction cost (see [2],[4]). The algorithm is based on linear underestimations to objective functions, and is successively used to solve large scale optimal portfolio selection problems with 200 assets and 60 simulated scenarios based on the MAD model (see [2]).

In this paper, we will propose a new branch and bound method to solve the large scale separable concave programming problems. A rectangular subdivision process (largest distance bisection, **LDB**) is proposed. Linear overestimations to objective $f(x)$ are employed to replace $f(x)$, and problems

$$\max\{g^i(x)|x \in V,\ l^i \leq x \leq u^i\} \qquad (LOP)$$

are successively solved, where the objective function $g^i(x) = \sum_{j=1}^n g_j^i(x_j)$, and $g_j^i(x_j),\ j = 1, 2, \ldots, n$, are linear overestimations to functions $f_j(x_j)$ over the set $S_i = [l^i, u^i]$.

Rectangle subdivision processes play an important role in branch and bound methods. As will be seen later, the concept of "normal rectangular subdivision" introduced in reference (3) will be concerned in the method proposed in this paper. The class of normal subdivision methods includes the exhaustive bisection, ω-subdivision and adaptive bisection (see [8]). A new bisection technique will be proposed and it will be shown that the proposed bisection method belongs to the class of normal rectangular subdivisions. Experiments on quadratic functions with different dimensions show that the iterative process with the proposed bisection technique can converge effectively. The quadratic function is a type concave function when we set its second order coefficients negative. The dimensions of tested problems ranges from 100 to 10000, large scale problems. The coefficients of these tested problems are randomly generated from uniform distribution. Tested functions are very important in economical and financial field, and usually used to express utilities or costs (see [5]). These functions often exhibit concave characteristics when they are used to denote cost functions or utility functions under the rule of margin cost (utility) decrease. When net returns or expected utilities are maximized, the resulting problem usually is a separable concave programming (see [7],[5],[2],).

A series of numerical experiments is presented and shows the efficiency of the proposed bisection method. It also shows that the algorithm can solve problems of practical size in an efficient way.

The rest of the paper is organized as follows. In Section 2, we will describe the new branch and bound algorithm. In section 3, we discuss the construction of a normal rectangular subdivision, and the largest distance bisection strategy will also be presented in this section. In section 4, we conduct a series of numerical tests, and present comparisons with some different bisection methods. Conclusions are given in section 5.

2. A Branch and Bound Algorithm

In this section, we describe the branch and bound method which bases upon a chosen normal rectangular subdivision process.

Let $S_0 = \{l_i \leq x_i \leq u_i,\ i = 1, 2, \ldots, n\}$ be a rectangle. We replace the convex functions $f_i(x_i)$ in $f(x)$ by an overestimated linear function $g_i^0(x_i)$ over S_0(see Figure 1),

$$g_i^0(x_i) = \delta_i x_i + \eta_i, \quad i = 1, 2, \ldots, n \tag{2.1}$$

where

$$\delta_i = \frac{f_i(u_i) - f_i(l_i)}{u_i - l_i}, \quad \eta_i = f_i(l_i) - \delta_i l_i, \quad i = 1, 2, \ldots, n. \tag{2.2}$$

Let

$$g^0(x) = \sum_{i=1}^n g_i^0(x_i),$$

then $g^0(x)$ is the convex envelope of the function $f(x)$ over the set S_0. We solve the linear

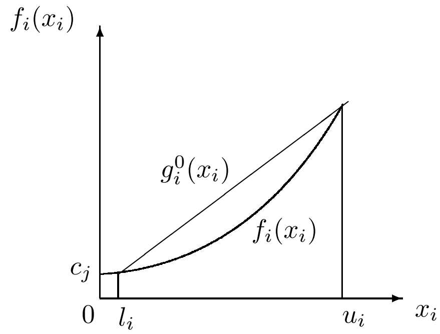

Figure 1: The overestimated linear function

overestimated approximation to (SCP),

$$\max\{g^0(x) = \sum_{i=1}^{n} g_i^0(x_i) | x \in V,\ l \le x \le u\} \tag{$\overline{Q}_0$}$$

$(\overline{Q}_0)$ is a linear programming problem. We use an interior point method to solve $(\overline{Q}_0)$ with large scale. Let $\bar{x}^0$ be an optimal solution of $(\overline{Q}_0)$, then we obtain an upper bound $g^0(\bar{x}^0)$ and a lower bound $f(\bar{x}^0)$ of the optimal value $f(x^*)$ of the problem (SCP) according to the following theorem.

Theorem 2.1. *Let $\bar{x}^0$ be an optimal solution of problem $(\overline{Q}_0)$ and x^* be a global optimal solution of problem (SCP). Then the following relation holds*

$$g^0(\bar{x}^0) \ge f^* \ge f(\bar{x}^0) \tag{2.3}$$

where $f^* = f(x^*)$.

Proof. It follows from $g^0(x) \ge f(x), \forall x \in [l, u]$ that

$$\begin{aligned} g^0(\bar{x}^0) &= max\{g^0(x) | x \in V, l \le x \le u\} \\ &\ge max\{f(x) | x \in V, l \le x \le u\} \\ &= f^* \ge f(\bar{x}^0). \end{aligned}$$

This gives the conclusion.

Rectangular subdivision processes can be used to subdivide a rectangle into a series of sub-rectangles by means of hyperplane parallel to certain facets. The generation of the family of sub-rectangles can be represented by a tree with root S_0 and nodes. A node is a successor of another one if and only if it represents an element of the latter node. An infinite path in the tree corresponds to an infinite nested sequence of rectangles $S_k,\ k = 0, 1, \ldots$.

Definition 2.1. *(*Horst and Tuy [3]*) Suppose $g^k(x) = \sum_{i=1}^n g_i^k(x_i)$ is the linear approximation to $f(x)$ over $S_k = \{x \mid l_i^k \le x_i \le u_i^k\}$ such that $g_i^k(l_i^k) = f_i(l_i^k)$ and $g_i^k(u_i^k) = f_i(u_i^k)$, $\bar{x}^k$ is the optimal solution of the problem that maximizes the function $g^k(x)$ over the intersection of V and S_k. A nested sequence S_k is said to be normal if*

$$\lim_{k\to\infty} |g^k(\bar{x}^k) - f(\bar{x}^k)| = 0 \tag{2.4}$$

A rectangular subdivision process is said to be normal if any nested sequence of rectangles generated from the process is normal.

We shall discuss some variants of a normal rectangular subdivision(**NRS**) process in the next section.

Theorem 2.1 indicates that if

$$g^0(\bar{x}^0) - f(\bar{x}^0) \leq \epsilon, \tag{2.5}$$

is satisfied with a given tolerance ϵ, then $\bar{x}^0$ is an approximate solution of (**SCP**) with error less than ϵ. If (2.5) does not hold, we will use an **NRS** process to divide the problem (**SCP**) into two subproblems:

$$\max\{f(x) = \sum_{i=1}^{n} f(x_i) | x \in V,\ x \in S_1\} \tag{Q_1}$$

and

$$\max\{f(x) = \sum_{i=1}^{n} f(x_i) | x \in V,\ x \in S_2\} \tag{Q_2}$$

where the sub-rectangles S_1 and S_2 are generated from S_0,

$$S_1 = \{x | l_s \leq x_s \leq h_s,\ l_j \leq x_j \leq u_j,\ j = 1, 2, \ldots, n,\ j \neq s\}, \tag{2.6}$$

$$S_2 = \{x | h_s \leq x_s \leq u_s,\ l_j \leq x_j \leq u_j,\ j = 1, 2, \ldots, n,\ j \neq s\}, \tag{2.7}$$

Using a similar way, we can get an overestimated linear programming to each of the branched subproblems (Q_1) and (Q_2) by replacing the function $f(x)$ with new overestimated linear functions $g^1(x)$ and $g^2(x)$, where

$$g^1(x) = \sum_{i \neq s} g_i^0(x_i) + g_s^1(x_s) \tag{2.8}$$

and

$$g^2(x) = \sum_{i \neq s} g_i^0(x_i) + g_s^2(x_s). \tag{2.9}$$

The following is the overestimated linear programming to problem (Q_1),

$$\max\{g^1(x) | x \in V,\ x \in S_1\} \tag{$\overline{Q}_1$}$$

If problem $(\overline{Q}_1)$ is infeasible, then problem (Q_1) is also infeasible, and we will delete problem (Q_1). Otherwise, let $\bar{x}^1$ be an optimal solution of $(\overline{Q}_1)$, then we obtain an upper bound $g^1(\bar{x}^1)$ and a low bound $f(\bar{x}^1)$ for the optimal value $f(x^1)$ of problem (Q_1) where x^1 is an optimal solution of (Q_1),. If $g^1(\bar{x}^1) < f(\bar{x}^0)$, then $f(x^1) \leq g^1(\bar{x}^1) < f(\bar{x}^0)$ (according to theorem 2.1), and problem (Q_1) will be deleted from further consideration. Otherwise, if $g^1(\bar{x}^1) - f(\bar{x}^1) < \epsilon$, then problem (Q_1) is solved with $\bar{x}^1$ being an approximate solution, if $f(\bar{x}^1) \geq f(\bar{x}^0)$, $\bar{x}^1$ will replace $\bar{x}^0$ as an approximation to the optimal solution. If $g^1(\bar{x}^1) - f(\bar{x}^1) \geq \epsilon$, the set S_1 will further be divided into two subsets to generate two subproblems using above **NRS** process as (2.6) and (2.7), Repeat this process until no subproblems exists.

Now we give the detailed description of the proposed branch-and-bound algorithm.

Algorithm 1. The Branch-and-Bound Algorithm

Step 0. *Set $k = 0$, $l^0 = l$, $u^0 = u$, give $\epsilon > 0$.*
Solve $(\overline{Q}_0)$ to get the optimal solution $\bar{x}^0$, and set $Q = \phi$.
*If $g^0(\bar{x}^0) - f(\bar{x}^0) \leq \epsilon$, then $\widetilde{x} = \bar{x}^0$, $\widetilde{f} = f(\bar{x}^0)$, go to **Step 6**; otherwise go to **Step 5**;*

Step 1. *Select a problem from **Q** and set it as (Q_k):*

$$max\{f(x) | x \in V,\ x \in S_k\} \tag{Q_k}$$

$\boldsymbol{Q} = \boldsymbol{Q} \backslash (Q_k)$; and solve the linear overestimated problem $(\overline{Q}_k)$:

$$max\{g^k(x) | x \in V,\ x \in S_k\}. \tag{$\overline{Q}_k$}$$

If $\overline{Q}_k$ is infeasible, go to **Step 6***, else let the optimal solution be $\bar{x}^k$.*

Step 2. *If $g^k(\bar{x}^k) \leq \widetilde{f}$, go to* **Step 6**.

Step 3. *If $g^k(\bar{x}^k) - f(\bar{x}^k) > \epsilon$, go to* **Step 5**;

Step 4. *If $f(\bar{x}^k) \geq \widetilde{f}$, set $\widetilde{x} = \bar{x}^k$, $\widetilde{f} = f(\widetilde{x})$;*

Step 5. *Divide S_k into $S_{k,1}$ and $S_{k,2}$ according to a chosen **NRS** process, generate two subproblems and place them into **Q**;*

Step 6. *If $\boldsymbol{Q} \neq \phi$ then set $k = k + 1$ and go to* **Step 1**, *else terminate with $\widetilde{x}$ being an ϵ optimal solution of (Q_0).*

Theorem 2.2. *The sequence $\widetilde{x}$ generated by the algorithm above converges to an optimal solution of (**SCP**) as $k \to \infty$.*

The proof of this theorem is similar to the proof of Theorems 3.18 in [6] and the proof of Theorem 3 in [4] and hence is omitted here.

3. Normal Rectangle Subdivisions

Various **NRS** processes for rectangle subdivision are available (see [8]). In this section, we present at first three bisection strategies which are often used, and then propose a new bisection technique called the largest distance bisection(**LDB**). Suppose that a rectangle $S_k = \{x \mid l_i^k \leq x_i \leq u_i^k,\ i = 1, 2, \cdots, n\}$ is selected for further division in Alg 1. For simplicity, we will denote the two sub-rectangles obtained from a bisection method as S_{+1}, S_{+2}, that is

$$S_{+1} = \{x \mid l_s^k \leq x_s^k \leq h_s,\ \ l_j^k \leq x_j^k \leq u_j^k,\ j = 1, 2, \ldots, n,\ j \neq s\}, \tag{3.1}$$

$$S_{+2} = \{x \mid h_s \leq x_s^k \leq u_s^k,\ \ l_j^k \leq x_j^k \leq u_j^k,\ j = 1, 2, \ldots, n,\ j \neq s\}. \tag{3.2}$$

Different choices of the index s and the point h_s gives different bisection techniques.

Exhaustive bisection

When the bisection index s is determined using the following rule

$$(u_s^k - l_s^k)^2 = max\{(u_i^k - l_i^k)^2,\ i = 1, 2, \ldots, n\},$$

and $h_s = (l_s^k + u_s^k)/2$, we get the well-known exhaustive bisection technique (see [8],[9]). That is, the middle point of the longest edge of the rectangle is selected to divide the rectangle into two sub-rectangles. It has been shown that any nested sequence of rectangles generated with the exhaustive bisection converges to a single point.

ω-subdivision

With the ω-subdivision technique, the bisection index s is determined using the following rule

$$g_s^k(\bar{x}_s^k) - f_s(\bar{x}_s^k) = max\{g_i^k(\bar{x}_i^k) - f_i(\bar{x}_i^k),\ i = 1, 2, \ldots, n\}, \tag{3.3}$$

$h_s = \bar{x}_s^k$ (see [8],[2]), where $\bar{x}^k$ is the optimal solution of problem $(\overline{Q}_k)$. The index s gives the greatest difference between $g_i^k(\bar{x}_i^k)$ and $f_i(\bar{x}_i^k)$.

Adaptive bisection

In the adaptive bisection technique, the index s is determined from

$$|\nu_s^k - \bar{x}_s^k| = \max_i |\nu_i^k - \bar{x}_i^k|,$$

$h_s = (\nu_s^k + \bar{x}_s^k)/2$, where $\bar{x}^k$ is the optimal solution of problem $(\overline{Q}_k)$ and $\nu_i^k \in argmin\{f_i(l_i^k), f_i(u_i^k)\}$.

It has been shown in [8],[9] that all the exhaustive bisection, ω-subdivision rules and adaptive bisection are normal rectangular subdivision processes.

At the rest of this section, we will propose the largest distance bisection (**LDB**) method and show that the proposed bisection is also a **NRS** process.

In the ω-subdivision technique, the index s is determined from the largest distance between $g_i^k(\bar{x}_i^k)$ and $f_i(\bar{x}_i^k)$ at the point $\bar{x}^k$ that is the optimal solution of problem $(\overline{Q}_k)$. With the largest distance bisection, the index s is determined by the largest difference between $g_i^k(x_i)$ and $f_i(x_i)$ over the subrectangle S_k. In order to determine such an index, we need to find the largest distance between $g_i^k(x_i)$ and $f_i(x_i)$ over the interval $[l_i^k, u_i^k]$ for all $i = 1, 2, \cdots, n$ and then find the index s from the n largest distance. Based on this idea, the largest bisection can be described as follows.

Algorithm 2. Largest Distance Bisection(LDB)

Step 1. *Calculate the slope of line of the overestimation function* $g_i^k(x_i)$

$$\delta_i = \frac{f_i(u_i^k) - f_i(l_i^k)}{u_i^k - l_i^k}, \quad i = 1, 2, \ldots, n$$

Express the distance between $g_i^k(x_i)$ *and* $f_i(x_i)$ *for* $x_i \in [l_i^k, u_i^k]$

$$d_i(x_i) = g_i^k(x_i) - f_i(x_i) = \delta_i x_i + \eta_i - f_i(x_i), \quad i = 1, 2, \ldots, n.$$

Step 2. *Maximize the distance function* $d_i(x_i)$ *to get the solution* h_i^k. *Let* $d_i(h_i^k)$ *be the maximum.*

Step 3. *Determine the index* s *from*

$$d_s(h_s^k) = max\{d_i(h_i^k),\ i = 1, 2, \ldots, n\}. \tag{3.4}$$

Step 4. *Determine the bisection point* h_s^k. *One method is to set* $h_s^k = \bar{x}_s^k$, *the other is to set* h_s^k *as the point calculated at* ***Step 2***.

In first case in **Step 4**, ω-subdivision is a special case of the largest distance bisection; In second case, the bisection point h_s^k is the tangent point of the line parallel to the line of the linear overestimate function $g_s^k(x_s)$ (see Figure 2).

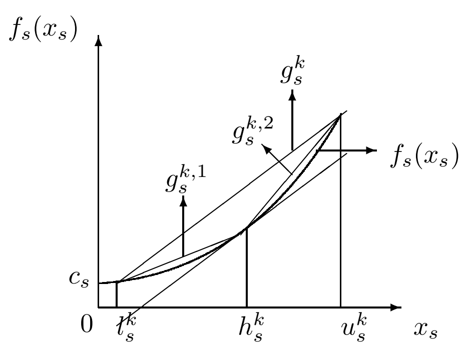

Figure 2: New bisection scheme

We give following lemma 3.3 to insure that the bisection point h_s^k is neither near left edge nor near right edge.

Lemma 3.3. *Suppose* S^k *is an infinite nested sequence of rectangles,* $k = 0, 1, 2, \ldots$, h_j^k *is the tangent point of the line parallel to the line of the linear overestimate function* $g_j^k(x_j)$, $j = 1, 2, \ldots, n$, h_s^k *is the bisection point calculated by the second method at* ***Step 4*** *in Alg 2, then*

$$h_s^k \geq l_s^k + \varepsilon'(u_s^k - l_s^k), \tag{3.5}$$

and

$$h_s^k \leq u_s^k - \varepsilon'(u_s^k - l_s^k), \tag{3.6}$$

holds. Where $0 < \varepsilon' \leq \frac{\varepsilon}{2nL_s\eta_s}$, ε *is the tolerance given at* ***Step 0*** *in Alg 1,* L_s *is given by (3.7) and* η_s *is the positive constant.*

Proof. Suppose (3.5) is not hold, then

$$h_s^k - l_s^k < \varepsilon^{'}(u_s^k - l_s^k),$$

Since $f_s(x_s)$ is Lipschitzian in any bounded interval([10], Theorem 10.4), then

$$|g_s^k(h_s^k) - f_s(h_s^k)| = |f_s(l_s^k) + \frac{f_s(u_s^k) - f_s(l_s^k)}{u_s^k - l_s^k}(h_s^k - l_s^k) - f_s(h_s^k)| \le 2\eta_s(h_s^k - l_s^k),$$

where η_s is a positive constant. Let $h^k = (h_1^k, h_2^k, \ldots, h_n^k)^T$,

$$L_s = h_s^k - l_s^k, \tag{3.7}$$

then

$$|g_s^k(h_s^k) - f_s(h_s^k)| < 2L_s\eta_s\varepsilon^{'} \le \frac{\varepsilon}{n}.$$

Since $g_s^k(h_s^k) - f_s(h_s^k) = \max_i\{g_i^k(h_i^k) - f_i(h_i^k)\}$, then we have

$$|g^k(h^k) - f(h^k)| \le \varepsilon.$$

This show that h^k is the approximate solution of the subproblem Q_k according to Alg 1, so rectangles $S_j,\ j > k$ does not exist. This result contrary to lemma's suppose that S_k is a infinite nested sequence of rectangles, so (3.5) hold.

Similarly, we also can show that (3.6) hold.

This lemma shows that when the algorithm **LDB** is used to determine the bisection point h_s^k, then (3.5) and (3.6) will hold with $\varepsilon' > 0$ sufficiently small. The following lemma will be used in the proof of the theorem 3.3 that shows **LDB** a **NRS** process.

Lemma 3.4. *Suppose S^k is an infinite nested sequence of rectangles, $k = 0, 1, 2, \ldots$, and h_j^k is the tangent point of the line parallel to the line of the linear overestimate function $g_j^k(x_j)$, $j = 1, 2, \ldots, n$. Let*

$$d_{max}^k = \max_i\{d_{max_i}^k\}, \tag{3.8}$$

and

$$d_{max_i}^k = \max_{x_i \in [l_i^k, u_i^k]}\{d_i(x_i)\} = g_i^k(h_i^k) - f_i(h_i^k). \tag{3.9}$$

If we select the bisection index s and bisection point h_s^k according to LDB method, then exist a subsequence $d_{max}^{k_t}$ of d_{max}^k convergence to zero when $t \to \infty$.

Proof. Because the nested sequence of rectangles is infinite and the edges of any rectangle are finite (n), there is an edge of the rectangle S^k that is divided infinitely. Thus we obtain an infinite nested interval sequence, say $[l_s^{k_t}, u_s^{k_t}],\ t = 1, 2, \ldots$. Suppose the bisection point of $[l_s^{k_t}, u_s^{k_t}]$ is $h_s^{k_t}$ at k_tth iteration, and consider the subinterval $[l_s^{k_{t+1}}, u_s^{k_{t+1}}]$ with $l_s^{k_{t+1}} = l_s^k,\ u_s^{k_{t+1}} = h_s^k$ (see Figure 3). Let $h_s^{k_{t+1}}$ be the bisection point at the k_{t+1}th iteration. Let $M_1,\ M_2,\ N_1,\ N_2,\ N_3$ and P be points given in Figure 3. Since $M_1M_2 = d_{max}^{k_t}$, we have $N_1N_3 \le M_1M_2$, and

$$d_{max}^{k_{t+1}} = N_1N_2 = N_1N_3 - N_2N_3 \le M_1M_2 - N_2N_3.$$

From the triangle PM_1M_2 we have

$$\frac{N_2N_3}{M_1M_2} = \frac{h_s^{k_{t+1}} - l_s^{k_t}}{h_s^{k_t} - l_s^{k_t}}$$

i.e.

$$N_2N_3 = M_1M_2\frac{h_s^{k_{t+1}} - l_s^{k_t}}{h_s^{k_t} - l_s^{k_t}}.$$

Then from (3.5) and (3.6) we obtain

$$h_s^{k_{t+1}} - l_s^{k_t} \ge \varepsilon^{'}(h_s^{k_t} - l_s^{k_t}).$$

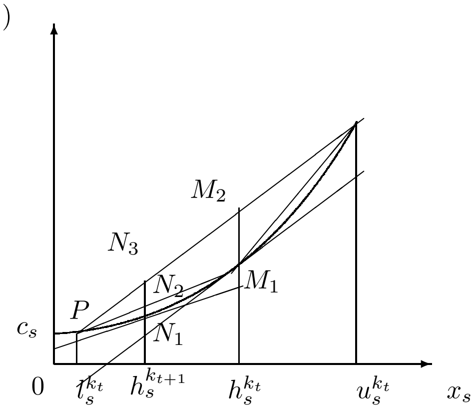

Figure 3: Largest Distant Bisection

Hence, we got

$$d_{max}^{k_{t+1}} \le N_1N_2 - N_2N_3 \le (1-\varepsilon^{'})M_1M_2 = (1-\varepsilon^{'})d_{max}^{k_t},$$

where $\varepsilon' > 0$ is a small constant. This inequality holds for all $t = 1, 2, \cdots$, and we have

$$d_{max}^{k_t} \to 0, \qquad when \quad t \to \infty.$$

This completes the proof.

The following theorem shows that the largest distance bisection also is an **NRS** process whenever the bisection point is given from these two possibilities.

Theorem 3.5. *The largest distance bisection process is a normal rectangular subdivision process.*

Proof. Suppose $S_k,\ k = 1, 2, ...,$ is a nested sequence of rectangles generated by the largest distance bisection process. Let the bisection index be s, and the bisection point $h_s \in [l_s^k,\ u_s^k]$.

If we select the bisection point as $h_s^k = \bar{x}_s^k$, then **LDB** is a **NRS** process (see [3], Proposition VII 17).

For the another choice of the bisection point, it is clear that $\{d_{max}^k\}$ is a monotone decreasing sequence. Lemma 3.4 indicates that there is a subsequence $d_{max}^{k_t}$ of d_{max}^k that converges to zero, thus we have

$$d_{max}^k \to 0, \text{ and } g_i^k(\bar{x}_i^k) - f_i(\bar{x}_i^k) \to 0, \quad \text{when} \quad k \to \infty. \tag{3.10}$$

which shows

$$|g^k(\bar{x}^k) - f(\bar{x}^k)| \to 0, \quad \text{when} \quad k \to \infty.$$

This show that the largest distance bisection is a **NRS** process, and the proof is completed.

4. Numerical Tests

We conducted numerical experiments of the proposed algorithm on the following quadratic functions with dimensions from 100 to 10000, large scale problems. As we pointed out that these problems occurs very often in economics and financial fields. We compare the proposed largest distance bisection with the available normal subdivision process. Because the ω-subdivision is more efficient than the exhaustive bisection and adaptive bisection (see[8]), we only compare the largest distance bisection (**LDB**) with the ω-subdivision. The program was coded by MatLab and tested on Pentium Pro 1794MHZ with 256 Mbyte memory. The parameter value $\varepsilon = 10^{-8}$ is used to terminate the iteration for tests on both the subdivisions. We used the breadth first rule for choosing subproblems in **step 1** at Alg 1.

Because the quadratic function is a type concave function when we set the coefficient of the second-order negative, tests are made on the following quadratic function with different dimensions,

$$f(x) = \sum_{i=1}^{n} f_i(x_i),$$

with constraints

$$V = \{x|x_1 + \ldots + x_n = 1\}, \quad \text{and} \quad l_i \le x_i \le u_i,\ l_i = 0,\ u_i = 1,\ i = 1, 2, \ldots, n, \tag{4.1}$$

where

$$f_i(x_i) = \frac{1}{2}a_i x_i^2 + b_i x_i + c_i, \qquad i = 1, 2, \ldots, n.$$

The coefficients $a_i(> 0)$, b_i and c_i are randomly generated in regions $[1, 2]$, $[-1, 1]$, $[0, 1]$ respectively in uniform distribution. The function $f(x)$ is convex, and the problem with maximizing $f(x)$ is a concave type optimization problem.

Table presents the numerical results for the problem with dimension $n = 1000$. In the table, the row titled with ω-subdivision gives the result obtained using the ω-subdivision technique to divide rectangles. LDB denotes the largest distance bisection (**LDB**) with 'Meth1' being $h_s^k = \bar{x}_s^k$, and 'Meth2' denoting that h_s^k is the second case given in **Step 4** of Alg 2. Every problem is tested in 10 times, and the numbers of minimal, maximal, average iterations (Min iter, Max iter, Avg iter) and mean CPU times (Time) are given in tables. It can be observed from Table 1 that the proposed largest bisection technique is efficient.

Table 1. Quadratic function numerical results

Bisection rule		Min iter	Avg iter	Max iter	Time(seconds)
ω -subdivision		1	3.8	37	35.0816
LDB	Meth1	1	2.6	18	18.4734
	Meth2	1	1.8	5	16.5971

Table gives the numerical results for different dimensions with the largest distance bisection and the bisection point being determined by Meth2.

Table 2. Different dimension numerical tests using LDB-Meth2

Dimension	Min iter	Avg iter	Max iter	Time(seconds)
100	1	4	19	12.2735
500	1	2.6	17	12.1770
1000	1	1.8	5	11.9957
2000	1	8.4	47	27.5518
3000	1	3.2	9	28.6567
4000	1	10	41	102.1004
5000	1	7.8	29	82.8255
6000	1	5	31	70.2896
7000	1	11	49	389.3896
8000	1	19.2	137	342.4349
9000	1	3.2	5	153.6293
10000	1	3.4	9	204.2293

Figure 4 plots the maximal, minimal and average CPU time for different dimension problems, where the average CPU time is plotted by the symbol "*", the CPU time from minimal to maximal is plotted by the vertical lines. It can also be observed from the figure that the proposed bisection technique is efficient for large scale separable concave programming problems. At the same time, we also test different dimension problems using ω-subdivision method and plot the average time on Figure 4. The efficiency of the LDB process can be observed from the figure.

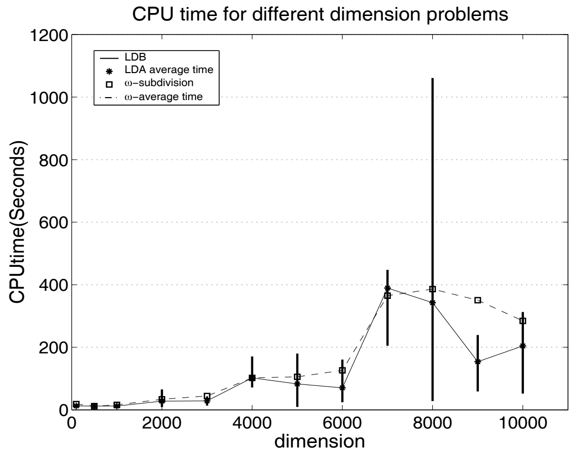

Figure 4: CPU time for different dimension problems

5. Conclusion

In this paper, we proposed a new branch and bound algorithm for large scale separable concave programming problems. The largest distance bisection(**LDB**) process is proposed to branch rectangles, and it is proved that **LDB** is a normal rectangular bisection (**NRS**) process. Numerical tests show that the new branch and bound algorithm is an effective method for large scale problems, and comparisons show that **LDB** process is more efficient than ω-subdivision. In practice, global optimization of a difference of two convex functions is also an important problem, which be named D-C programming in literature. We will show that the proposed bisection technique can also solving large scale D-C programming problems effectively when the convex parts of the objective function is quadratic and feasible region is a polytope with rectangular constraints. We will present this application in later paper.

References

[1] Falk J.E. and Soland R.M., An algorithm for separable nonconvex programming problems, *Manage. Sci.*, **15** (1969), 550-569.

[2] Hiroshi Konno, and Annista Wijayanayake, Portfolio Optimazation Problem under Concave Transaction Costs and Minimal Transaction Unit Constrains, *Math. Program.*, Ser.B, **89** (2001), 233-250.

[3] Horst R. and Tuy H., Global Optimization Deterministic Approaches, 3nd Edition, Springer, Berlin, 1997.

[4] Konno H., Thach P. T. and Tuy H., Optimization on Low Rank Nonconvex Structures, Kluwer Academic Publishers, 1997.

[5] Markowitz H. M., Portfolio Selection, *Jounal of Finance*, **7**:1 (1952), 77-79.

[6] Reiner Horst, Panos M. Pardalos and Nguyen V. Thoai, Introduction to Globle Optimization, Kluwer Academic Publishers, London, 1995.

[7] Steinbach M.C., Markowitz Revisited: Single-Period and Multi-Period Mean-Variance Models, working Paper, Konrad-Zentrum für Informationstechnik Berlin, Preprint SC-99-30, Aug, 1999.

[8] Thai Quynh Phong, Le Thi Hoai An, and Pham Dinh Tao, Decomposition Branch and Bound Method for Globally Solving Linear Constrained Indefinate QPP, *Operations Research Letters,* **17** (1995), 215-220.

[9] Tuy H., Convex Analysis and Global Optimization, Kluwer Academic Publishers, 1998.

[10] Rockafellar R.T., Convex Analysis, Princeton University Press, Princeton, New Jersey, 1970.