A Simplicial Algorithm for Concave Minimization and Its Performance as a Heuristic Tool

Takahito Kuno[*] and Yoshiyuki Shiguro[†]

Graduate School of Systems and Information Engineering

University of Tsukuba, Tsukuba, Ibaraki 305-8573, Japan

July 2007

Abstract

In this paper, we develop a kind of branch-and-bound algorithm for solving concave minimization problems. We show that the algorithm converges to an optimal solution of this multiextremal global optimization problem, and that it generates a high-quality heuristic solution even if it is forced to terminate. Therefore, the algorithm can be used in two ways, as an exact algorithm and as a heuristic tool. We also report some numerical results of a comparison with an existing algorithm, and show the performance as a heuristic tool.

Key words: Global optimization, concave minimization, branch-and-bound algorithm, simplicial algorithm, heuristic algorithm.

1 Introduction

Since the pioneer work of Tuy [17], various algorithms have been proposed to solve concave minimization problems, where a concave function f is minimized globally on a polyhedral subset D of $\mathbb{R}^n$. If the objective function f is linear, the problem is just a linear program and can be solved in polynomial time [2]. In the nonlinear case, however, the only clue to a globally optimal solution is that it exists among vertices of the feasible set D. Since the maximum number of vertices of D is exponential in n [3], it is easy to imagine how difficult the problem is to solve. In fact, it is known to be NP-hard in the sense of computational complexity, even when f is quadratic and D is a box [16].

The most popular method for solving this global optimization problem is the branch-and-bound, commonly used in solving integer programs. However, while the latter

[*]E-mail: takahito@cs.tsukuba.ac.jp

[†]Current affiliation: Software Knowledge Incorporated, Tokyo 107-0052, Japan

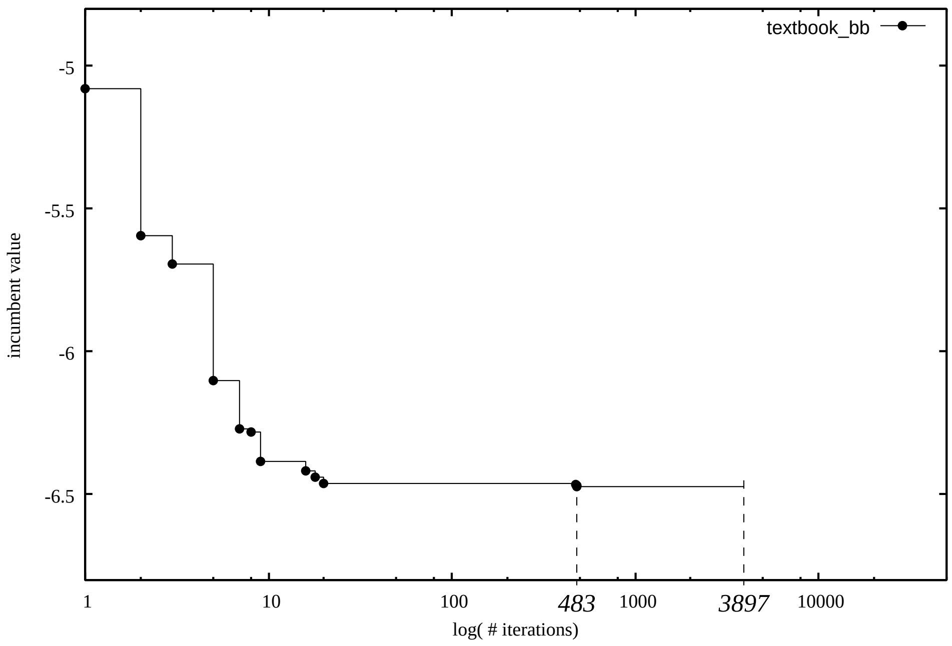

Figure 1: Changes in the incumbent value in the simplicial algorithm.

reduces to a set of obvious subproblems as branching continues, the concave minimization problem does not in general. As a consequence, the branch-and-bound algorithm often exhibits long-tail convergence, as is shown in Figure 1. This figure demonstrates how the incumbent value is updated while a concave quadratic minimization problem of 20 nonlinear variables is solved using a simplicial algorithm [4], a variant of the branch-and-bound algorithm contained in any textbook on global optimization (see e.g., [5, 8, 18]). The algorithm takes $3,897$ iterations for this particular instance, but it actually obtains a globally optimal solution in 483 iterations. All the rest of the iterations are spent only in checking its optimality.

It is true that many applications require a globally optimal solution in high-precision, even at the expense of long-tail convergence as in the above branch-and-bound algorithm. Meanwhile, there should also be many applications where a good feasible solution is enough if it is available in a small amount time. In this paper, to balance those competing needs, we develop a new kind of branch-and-bound algorithm for solving the concave minimization problem. We show that the algorithm converges to a rigorous optimal solution, and besides that it generates a high quality heuristic solution if it is forced to terminate in the early stage of iterations. In other words, the proposed algorithm can be used in two ways, as an exact algorithm and as a heuristic tool. In the next section, we will explain the basic workings of the usual simplicial algorithm, which provides a framework for our algorithm. In Section 3, we will make necessary changes and improvements to this algorithm in such a way as (i) to save efforts in the bounding operation, (ii) to strengthen the upper bound (incumbent value), not the lower bound, and (iii) to preserve the convergence property. In Section 4, we will present a detailed description of the algorithm incorporating these three devices and a convergence proof. In Section 5, we will report some numerical results of a comparison of the algorithm with an existing one, and show the performance as a heuristic tool. Lastly, we will summarize the paper in Section 6, with some concluding remarks.

2 Simplicial algorithm for concave minimization

Let f be a concave function defined on an open subset of $\mathbb{R}^n$ including a polyhedron:

$$D = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\},$$

where $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{b} \in \mathbb{R}^m$. The problem considered in this paper is a minimization of f over D, i.e.,

$$\left|\begin{array}{ll} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{x} \in D. \end{array}\right. \tag{1}$$

We assume that f is continuously differentiable, D is nonempty and bounded. Therefore, (1) has a globally optimal solution among veritces of D, though many of those are just locally optimal.

One of the most popular methods for locating a globally optimal solution is the simplicial branch-and-bound algorithm, originally proposed in [4]. In this algorithm, we first define an n-simplex Δ^1 including D. The domain of f is assumed to be large enough to include this simplex. Let $\mathbf{v}_1^1, \ldots, \mathbf{v}_{n+1}^1$ denote the vertices of Δ^1. Then we have

$$\Delta^1 = \operatorname{conv}(\{\mathbf{v}_1^1, \ldots, \mathbf{v}_{n+1}^1\}) \equiv \left\{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{x} = \mathbf{V}^1\boldsymbol{\lambda},\ \mathbf{e}^\mathsf{T}\boldsymbol{\lambda} = 1,\ \boldsymbol{\lambda} \geq \mathbf{0}\right\},$$

where $\mathbf{V}^1 = [\mathbf{v}_1^1, \ldots, \mathbf{v}_{n+1}^1]$, and $\mathbf{e} \in \mathbb{R}^{n+1}$ is the all-ones vector. As subdividing Δ^1 into smaller simplices $\{\Delta^k \mid k \in \mathcal{K}\}$ such that

$$\bigcup_{k\in\mathcal{K}} \Delta^k = \Delta^1, \quad \operatorname{int}(\Delta^r) \cap \operatorname{int}(\Delta^s) = \emptyset, \quad r \neq s,$$

we compute a lower bound $w(\Delta^k)$ of f on $D \cap \Delta^k$ for each $k \in \mathcal{K}$, where $\operatorname{int}(\,\cdot\,)$ denotes the set of interior points. We will discuss how to compute $w(\Delta^k)$ later in detail.

Suppose a feasible solution $\widetilde{\mathbf{x}} \in D$ is given. If $f(\widetilde{\mathbf{x}}) \leq w(\Delta^k)$, then we see that no $\mathbf{x} \in D \cap \Delta^k$ satisfies $f(\mathbf{x}) < f(\widetilde{\mathbf{x}})$. Therefore, if $f(\widetilde{\mathbf{x}}) \leq w(\Delta^k)$ for every $k \in \mathcal{K}$, we can conclude that $\widetilde{\mathbf{x}}$ is a globally optimal solution. If not, we subdivide Δ^k for some appropriate $k \in \mathcal{K}$ further into smaller simplices and replace Δ^k by those. The algorithm is outlined as follows:

Prototype of the simplicial algorithm. Compute a lower bound $w(\Delta^1)$ and let $\widetilde{\mathbf{x}}$ be a feasible solution found in the process. Let $\mathcal{L} := \{\Delta^1\}$, $k := 1$, and repeat Steps 1–3.

Step 1 (subproblem selection) Take a simplex Δ^k with the least value of w from $\mathcal{L}$.

Step 2 (bounding operation) If $f(\widetilde{\mathbf{x}}) \leq w(\Delta^k)$, then terminate. Otherwise, go to Step 3.

Step 3 (branching operation) Subdivide Δ^k into two subsimplices Δ_1 and Δ_2. Compute $w(\Delta_1)$ and $w(\Delta_2)$ and update the incumbent $\widetilde{\mathbf{x}}$ if possible. Add Δ_1 and Δ_2 to $\mathcal{L}$. Let $k := k + 1$. □

When there is no Δ with $w(\Delta) < f(\widetilde{\mathbf{x}})$ in the list $\mathcal{L}$, the current incumbent $\widetilde{\mathbf{x}}$ is a globally optimal solution of (1). In general, however, it does not happen, and the algorithm would generate an infinite nested sequence $\{\Delta^{k_q} \mid q = 1, 2, \dots\}$ such that

$$\Delta^{k_1} \supset \Delta^{k_2} \supset \cdots , \tag{2}$$

where Δ^{k_q} denotes the simplex selected at Step 1 of the k_qth iteration. If $\bigcap_{q=1}^{\infty} \Delta^{k_q}$ is a singleton for every infinite nested sequence, the algorithm is guaranteed to be convergent. The only known subdivision rule with such exhaustiveness is bisection. Let $\Delta^k = \mathrm{conv}(\{\mathbf{v}_1^k, \dots, \mathbf{v}_{n+1}^k\})$. At Step 3, we may select the longest edge, say $[\mathbf{v}_r^k, \mathbf{v}_s^k]$, and divide it at a fixed ratio of $\alpha \in (0, 1/2]$. Letting $\mathbf{v} = (1-\alpha)\mathbf{v}_r^k + \alpha\mathbf{v}_s^k$, we have

$$\Delta_1 = \mathrm{conv}(\{\mathbf{v}_i^k \mid i \neq r\} \cup \{\mathbf{v}\}), \quad \Delta_2 = \mathrm{conv}(\{\mathbf{v}_i^k \mid i \neq s\} \cup \{\mathbf{v}\}).$$

Another well-known subdivision rule is ω-subdivision, which subdivides Δ^k radially into $n+1$ subsimplices and has no exhaustiveness property. It has been noted for a long time that ω-subdivision is practically more efficient than bisection, but the convergence was a theoretical open question until 2000 (see [8, 13] for more details).

3 Linear relaxation and local search

In the usual simplicial branch-and-bound algorithm, the lower bound $w(\Delta)$ of f on $D \cap \Delta$ is computed by minimizing a convex envelope g of f on Δ, which is a maximal convex function underestimating f on Δ. In our case where f is concave, g is an affine function, which agrees with f at the $n+1$ vertices $\mathbf{v}_1, \dots, \mathbf{v}_{n+1}$ of Δ. By noting that each $\mathbf{x} \in \Delta$ can be written as $\mathbf{x} = \mathbf{V}\boldsymbol{\lambda}$ for some $\boldsymbol{\lambda} \geq \mathbf{0}$ such that $\mathbf{e}^{\mathsf{T}}\boldsymbol{\lambda} = 1$, we have the value of g at $\mathbf{x} \in \Delta$ as follows

$$g(\mathbf{x}) = \sum_{i=1}^{n+1} \lambda_i f(\mathbf{v}_i).$$

Thereby, $w(\Delta)$ is given as the optimal value of a linear program:

$$\mathrm{Q}(\Delta)\left|\begin{array}{ll}\text{minimize} & \displaystyle\sum_{i=1}^{n+1}\lambda_i f(\mathbf{v}_i)\\ \text{subject to} & \mathbf{AV}\boldsymbol{\lambda}=\mathbf{b},\quad \mathbf{V}\boldsymbol{\lambda}\geq\mathbf{0}\\ & \mathbf{e}^{\mathsf{T}}\boldsymbol{\lambda}=1,\quad \boldsymbol{\lambda}\geq\mathbf{0},\end{array}\right.$$

which is a linear relaxation of a subproblem of (1):

$$\mathrm{P}(\Delta)\left|\begin{array}{ll}\text{minimize} & f(\mathbf{x})\\ \text{subject to} & \mathbf{x}\in D\cap\Delta.\end{array}\right.$$

Although $\mathrm{Q}(\Delta)$ yields a fairly tight lower bound of $\mathrm{P}(\Delta)$ and $\Delta\in\mathcal{L}$ might be fathomed at an early iteration of the algorithm, our purpose is not necessarily to accelerate the convergence but to obtain a better feasible solution in a smaller amount of time. We will therefore try to simplify the relaxation of $\mathrm{P}(\Delta)$. This leads to a deterioration in the lower bound of $\mathrm{P}(\Delta)$, but has some redeeming merits, especially when the simplicial algorithm is used as a heuristic tool.

3.1 Simplified linear relaxation

Let us denote the centroid of Δ by $\mathbf{u}=\sum_{i=1}^{n+1}\mathbf{v}_i/(n+1)$ and define

$$h(\mathbf{x})=\nabla f(\mathbf{u})\mathbf{x}+\delta,$$

where

$$\delta=\min\{f(\mathbf{x})-\nabla f(\mathbf{u})\mathbf{x}\mid\mathbf{x}\in\Delta\}. \tag{3}$$

Note that the right-hand side of (3) is a concave minimization, but the feasible set Δ has only $n+1$ vertices, at least one of which gives δ. Let us replace f by h in $\mathrm{P}(\Delta)$ and further drop the constraint $\mathbf{x}\in\Delta$. Then we have another linear relaxation of $\mathrm{P}(\Delta)$:

$$\mathrm{R}(\Delta)\left|\begin{array}{ll}\text{minimize} & \nabla f(\mathbf{u})\mathbf{x}+\delta\\ \text{subject to} & \mathbf{A}\mathbf{x}=\mathbf{b},\quad \mathbf{x}\geq\mathbf{0}.\end{array}\right.$$

Let $z(\Delta)$, $w_Q(\Delta)$ and $w_R(\Delta)$ denote the optimal value of $\mathrm{P}(\Delta)$, $\mathrm{Q}(\Delta)$ and $\mathrm{R}(\Delta)$, respectively. The former two values are considered to be $+\infty$ if $D\cap\Delta=\emptyset$.

Lemma 3.1. *Among $z(\Delta)$, $w_Q(\Delta)$ and $w_R(\Delta)$ exists a relationship:*

$$w_R(\Delta)\leq w_Q(\Delta)\leq z(\Delta). \tag{4}$$

Proof. By definition, both objective functions g and h of $\mathrm{Q}(\Delta)$ and $\mathrm{R}(\Delta)$ underestimate f on Δ, but the former is a maximal underestimating function. Moreover, the feasible set $D \cap \Delta$ of $\mathrm{P}(\Delta)$ and $\mathrm{Q}(\Delta)$ is a subset of the feasible set D of $\mathrm{R}(\Delta)$. Hence, the relationship (4) is a natural consequence. $\square$

In terms of bound tightness, our proposed relaxation $\mathrm{R}(\Delta)$ is inferior to the usual $\mathrm{Q}(\Delta)$, as shown in Lemma 3.1. However, $\mathrm{R}(\Delta)$ has the following advantages:

(i) the objective function h is easy to obtain if gradient vectors of f are given analytically,

(ii) the feasible set D remains unchanged regardless of the variation of Δ, and

(iii) the structure of the target problem (1) is inherited to $\mathrm{R}(\Delta)$.

In the simplicial algorithm, we need to repeatedly solve the same type of relaxed problems. If we adopt $\mathrm{R}(\Delta)$ instead of $\mathrm{Q}(\Delta)$, this computational burden is reduced considerably because the optimal solution to the preceding relaxed problem is feasible for the current $\mathrm{R}(\Delta)$. We could recover its optimality for $\mathrm{R}(\Delta)$ in a very few simplex pivots. If the target problem (1) has some favorable structure as in minimum concave-cost network flow problems [6, 7, 12, 14], this burden would be further reduced. Certainly, if the convex envelope g is represented explicitly as a function in $\mathbf{x}$, we can give the properties (ii) and (iii) to $\mathrm{Q}(\Delta)$, just by dropping the constraint $\mathbf{x} \in \Delta$. Unfortunately, to obtain the explicit form of g, we have to solve a linear system of size $n + 1$, and besides its solution becomes numerically unstable as Δ grows smaller [9, 10].

3.2 Improving of the incumbent by local search

The last property (iii) also means that any optimal basic solution $\mathbf{x}^0$ of $\mathrm{R}(\Delta)$ is a feasible basic solution of the target problem (1) and might be a locally optimal solution. Even if not, we can reach a better solution, by visiting some adjacent vertices of D from $\mathbf{x}^0$.

Let $\mathbf{c} = \nabla f(\mathbf{x}^0)$ and consider a linear program associated with (1):

$$\left|\begin{array}{ll} \text{minimize} & \mathbf{cx} \\ \text{subject to} & \mathbf{Ax} = \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0}. \end{array}\right. \tag{5}$$

Let B denote the basis of $\mathbf{x}^0$, the index set of m basic variables in $\mathbf{x}^0$, and let $N = \{1, \ldots, n\} \setminus B$. Also denote by $(\mathbf{A}_B, \mathbf{A}_N)$, $(\mathbf{c}_B, \mathbf{c}_N)$ and $(\mathbf{x}_B, \mathbf{x}_N)$, respectively, the corresponding partitions of $\mathbf{A}$, $\mathbf{c}$ and $\mathbf{x}$. Then we have a feasible dictionary of (5):

$$\left|\begin{array}{lcl} \mathbf{x}_B & = & \mathbf{b}' - \mathbf{A}'_N \mathbf{x}_N \\ z & = & \mathbf{c}_B \mathbf{b}' + (\mathbf{c}_N - \mathbf{c}_B \mathbf{A}'_N)\mathbf{x}_N, \end{array}\right. \tag{6}$$

where $\mathbf{A}'_N = \mathbf{A}_B^{-1}\mathbf{A}_N$ and $\mathbf{b}' = \mathbf{A}_B^{-1}\mathbf{b}$. The basic solution $\mathbf{x}^0$ is given as $(\mathbf{x}_B^0, \mathbf{x}_N^0) = (\mathbf{b}', \mathbf{0})$ via this dictionary.

Proposition 3.2. *If $c_N - \mathbf{c}_B\mathbf{A}'_N > \mathbf{0}$, then $\mathbf{x}^0$ is a locally optimal solution of (1).*

Proof. Let $\mathbf{x}'$ be an arbitrary point in $D \setminus \{\mathbf{x}^0\}$. Then there exist a unit vector $\mathbf{d}$ and a positive number t such that $\mathbf{x}' = \mathbf{x}^0 + t\mathbf{d}$. Since $(\mathbf{x}'_B, \mathbf{x}'_N) = (\mathbf{b}' - t\mathbf{A}'_N\mathbf{d}_N, t\mathbf{d}_N)$, we have

$$\begin{aligned} f(\mathbf{x}') - f(\mathbf{x}^0) &= \nabla f(\mathbf{x}^0)(\mathbf{x}' - \mathbf{x}^0) + \|\mathbf{x}' - \mathbf{x}^0\|\beta(t) \\ &= t[(\mathbf{c}_N - \mathbf{c}_B\mathbf{A}'_N)\mathbf{d}_N + \beta(t)], \end{aligned}$$

where $\beta(t) \leq 0$ by the concavity of f and $\lim_{t\to 0}\beta(t) = 0$. If $\mathbf{d}_N \leq \mathbf{0}$, then $\mathbf{x}' \notin D$ or $\mathbf{x}' = \mathbf{x}^0$. Therefore, we may assume that $\mathbf{d}_N \geq 0$ and $\mathbf{d}_N \neq \mathbf{0}$. This implies $(\mathbf{c}_N - \mathbf{c}_B\mathbf{A}'_N)\mathbf{d}_N > 0$, and besides $(\mathbf{c}_N - \mathbf{c}_B\mathbf{A}'_N)\mathbf{d}_N + \beta(t) \geq 0$ for sufficiently small $t > 0$. Hence, we have $f(\mathbf{x}') - f(\mathbf{x}^0) \geq 0$ if $\mathbf{x}'$ is in some neighborhood of $\mathbf{x}^0$. $\square$

Suppose in the dictionary (6) that for some $s \in N$ we have

$$c_s - \mathbf{c}_B\mathbf{a}'_s < 0, \tag{7}$$

where $\mathbf{a}'_s$ denotes the corresponding column of $\mathbf{A}'_N$. Then we perform a simplex pivot in some appropriate row, say corresponding to an $r \in B$, of $\mathbf{a}'_s$. Let $\mathbf{x}^1$ denote the resulting feasible basic solution. Since f is concave, we have

$$f(\mathbf{x}^1) - f(\mathbf{x}^0) \leq \nabla f(\mathbf{x}^0)(\mathbf{x}^1 - \mathbf{x}^0) = \left(c_j - \mathbf{c}_B\mathbf{A}_B^{-1}\mathbf{a}_j\right)(x_s^1 - x_s^0). \tag{8}$$

If the pivot is nondegenerate, i.e., $x_s^1 - x_s^0 = x_s^1 > 0$, then (8), together with (7), implies $f(\mathbf{x}^1) < f(\mathbf{x}^0)$. After updating $B = B \cup \{s\} \setminus \{r\}$ and $N = N \cup \{r\} \setminus \{s\}$, we again replace $\mathbf{c} = \nabla f(\mathbf{x}^0)$ by $\mathbf{c} = \nabla f(\mathbf{x}^1)$, and perform a simplex pivot in the dictionary (6) with respect to the new (B, N) if possible. Continuing this process with the help of some cycling prevention rule for the simplex method (see e.g., [1]), we have a finite sequence of feasible basic solutions $\{\mathbf{x}^\ell \mid \ell = 0, 1, \ldots, p\}$ such that

$$f(\mathbf{x}^0) \geq f(\mathbf{x}^1) \geq \cdots \geq f(\mathbf{x}^p),$$

and the following holds in the dictionary (6) with $\mathbf{c} = \nabla f(\mathbf{x}^p)$:

$$\mathbf{c}_N - \mathbf{c}_B\mathbf{A}'_N \geq \mathbf{0}. \tag{9}$$

If the inequality holds strictly in (9), then Proposition 3.2 guarantees the local optimality of $\mathbf{x}^p$. Even if not, we could improve the incumbent $\widetilde{\mathbf{x}}$ with $\mathbf{x}^p$, except when all the pivots are degenerate.

4 Convergence and heuristic application

Let K be a predetermined number of iterations and $\epsilon \geq 0$ a tolerance for the value of f. Our proposed simplicial branch-and-bound algorithm is described as follows:

```
algorithm simplicial_bb
begin
    compute an $n$-simplex $\Delta^1 = \mathrm{conv}(\{\mathbf{v}_1^1, \dots, \mathbf{v}_{n+1}^1\})$ including $D$;
    let $\delta^1 := \min\{f(\mathbf{x}) - \nabla f(\mathbf{u}^1)\mathbf{x} \mid \mathbf{x} \in \Delta\}$ for the centroid $\mathbf{u}^1$ of $\Delta^1$;
    solve $\mathrm{R}(\Delta^1)$ of minimizing $h^1(\mathbf{x}) := \nabla f(\mathbf{u}^1)\mathbf{x} + \delta^1$ and obtain $w_R(\Delta^1)$;
    let $\mathbf{x}^1$ denote the feasible basic solution satisfying $h^1(\mathbf{x}^1) = w_R(\Delta^1)$;
    $\widetilde{\mathbf{x}}^1$ := local_search($\mathbf{x}^1$); $\widetilde{z}^1 := f(\widetilde{\mathbf{x}}^1)$; $\mathcal{L} := \{\Delta\}$; $k := 1$;
    while $k \leq K$ do begin                                              / * Step 1 * /
        select a simplex $\Delta^k = \mathrm{conv}(\{\mathbf{v}_1^k, \cdots, \mathbf{v}_{n+1}^k\})$ with the least value of $w_R$ from $\mathcal{L}$;
        let $h^k(\mathbf{x}) = \nabla f(\mathbf{u}^k)\mathbf{x} + \delta^k$ denote the objective function of $\mathrm{R}(\Delta^k)$;
        let $\mathbf{x}^k$ denote the feasible basic solution satisfying $h^k(\mathbf{x}^k) = w_R(\Delta^k)$; / * Step 2 * /
        if $\widetilde{z}^k - w_R(\Delta^k) \leq \epsilon$ then $k := K + 1$
        else begin
            subdivide $\Delta^k$ into $\Delta_1$ and $\Delta_2$ according to the bisection rule;    / * Step 3 * /
            for $i = 1, 2$ do begin
                let $\delta := \min\{f(\mathbf{x}) - \nabla f(\mathbf{u})\mathbf{x} \mid \mathbf{x} \in \Delta_i\}$ for the centroid $\mathbf{u}$ of $\Delta_i$;
                solve $\mathrm{R}(\Delta_i)$ of minimizing $h(\mathbf{x}) := \nabla f(\mathbf{u})\mathbf{x} + \delta$ and obtain $w_R(\Delta_i)$;
                let $\mathbf{x}_i$ denote the feasible basic solution satisfying $h(\mathbf{x}_i) = w_R(\Delta_i)$;
                $\widetilde{\mathbf{x}}_i$ := local_search($\mathbf{x}_i$)
            end;
            $\mathcal{L} := \mathcal{L} \cup \{\Delta_1, \Delta_2\}$
        end;
        select $\widetilde{\mathbf{x}}^{k+1} \in \arg\min\{f(\mathbf{x}) \mid \mathbf{x} = \widetilde{\mathbf{x}}_1, \widetilde{\mathbf{x}}_2, \widetilde{\mathbf{x}}^k\}$ and let $\widetilde{z}^{k+1} := f(\widetilde{\mathbf{x}}^{k+1})$;
        $\mathcal{L} := \mathcal{L} \setminus \{\Delta^k\}$; $k := k + 1$
    end;
    output $(\widetilde{\mathbf{x}}^k, \widetilde{z}^k)$
end;

function local_search($\mathbf{x}$)
begin
    let $B$ denote the basis for $\mathbf{x}$ and $N := \{1, \dots, , n\} \setminus B$;
    let $(\mathbf{c}_B, \mathbf{c}_N)$ and $(\mathbf{A}_B, \mathbf{A}_N)$ denote the partitions of $\mathbf{c} := \nabla f(\mathbf{x})$ and $\mathbf{A}$;
    $\mathbf{A}'_N := \mathbf{A}_B^{-1}\mathbf{A}_N$; $\mathbf{b}' := \mathbf{A}_B^{-1}\mathbf{b}$;
    while $\mathbf{c}_N - \mathbf{c}_B\mathbf{A}'_N \geq \mathbf{0}$ does not hold do begin
```

```
            select a column $\mathbf{a}'_s$ of $\mathbf{A}'_N$ such that $c_s - \mathbf{c}_B\mathbf{a}'_s < 0$;
            perform a pivot in an appropriate row of $\mathbf{a}'_s$, corresponding to an $r \in B$;
            $B := B \cup \{s\} \setminus \{r\}$; $N := N \cup \{r\} \setminus \{t\}$;
            update $(\mathbf{c}_B, \mathbf{c}_N)$, $(\mathbf{A}_B, \mathbf{A}_N)$, $\mathbf{A}'_N$ and $\mathbf{b}'$ according to the new $(B, N)$;
            Let $\mathbf{x}$ denote the basic solution with $(\mathbf{x}_B, \mathbf{x}_N) = (\mathbf{b}', \mathbf{0})$ and let $\mathbf{c} := \nabla f(\mathbf{x})$
        end;
        return $\mathbf{x}$
    end;
```

4.1 Convergence proof

If $K = +\infty$ and $\epsilon = 0$, the algorithm simplicial_bb does not terminate in general and generates an infinite nested sequence $\{\Delta^{k_q} \mid q = 1, 2, \dots\}$ as in (2), like the usual simplicial algorithm with the bisection rule. However, we can show that the difference between the incumbent value $\widetilde{z}^{k_q}$ and the lower bound $w_R(\Delta^{k_q})$ diminishes to zero in a subsequence of $\{\Delta^{k_q} \mid q = 1, 2, \dots\}$. This leads to the convergence of the algorithm simplicial_bb.

Lemma 4.1. *Let $\{\Delta^k \mid k \in K\}$ be any infinite nested sequence of simplices generated by the algorithm* simplicial_bb*. Then there exists a subsequence $\{k_1, k_2, \dots\} \subset K$ such that*

$$\lim_{q \to \infty} \left[\widetilde{z}^{k_q} - w_R(\Delta^{k_q})\right] = 0. \tag{10}$$

Proof. For every $q = 1, 2, \dots$, we can assume

$$f(\mathbf{x}^{k_q}) \geq \widetilde{z}^{k_q} > w_R(\Delta^{k_q}) = \nabla f(\mathbf{u}^{k_q})\mathbf{x}^{k_q} + \delta^{k_q}. \tag{11}$$

Let $\mathbf{v}^{k_q}$ denote the vertex of Δ^{k_q} defining δ^{k_q}. Then we have

$$\delta^{k_q} = f(\mathbf{v}^{k_q}) - \nabla f(\mathbf{u}^{k_q})\mathbf{v}^{k_q}.$$

Recall that we adopt the bisection rule and have $\{\overline{\mathbf{v}}\} = \bigcap_{k \in K} \Delta^k$. Then $\mathbf{u}^{k_q} \to \overline{\mathbf{v}}$ and $\mathbf{v}^{k_q} \to \overline{\mathbf{v}}$ as $q \to +\infty$, because $\mathbf{u}^{k_q}$ and $\mathbf{v}^{k_q}$ are points in Δ^{k_q}. By the continuity of f and ∇f, as $q \to +\infty$, we have

$$\nabla f(\mathbf{v}^{k_q}) \to \nabla f(\overline{\mathbf{v}}), \quad \delta^{k_q} \to f(\overline{\mathbf{v}}) - \nabla f(\overline{\mathbf{v}})\overline{\mathbf{v}}.$$

We can also assume $\mathbf{x}^{k_q} \to \overline{\mathbf{x}} \in D$ as $q \to +\infty$, because $\mathbf{x}^k$'s are generated in the compact set D. Therefore, as $q \to +\infty$, we have

$$w_R(\Delta^{k_q}) \to \nabla f(\overline{\mathbf{v}})(\overline{\mathbf{x}} - \overline{\mathbf{v}}) + f(\overline{\mathbf{v}}) \geq f(\overline{\mathbf{x}}),$$

by noting the concavity of f. This, together with (11), implies (10). □

Theorem 4.2. *Suppose $K = +\infty$ and $\epsilon = 0$. If the algorithm* simplicial_bb *terminates at the kth iteration, then $\widetilde{\mathbf{x}}^k$ is a globally optimal solution of (1). If not, every accumulation point of the sequence $\{\widetilde{\mathbf{x}}^k \mid k = 1, 2, \dots\}$ is a globally optimal solution.*

Proof. If simplicial_bb terminates, the assertion is obvious. Let us assume that it does not terminate. Then simplicial_bb generates at least one infinite nested sequence $\{\Delta^k \mid k \in K\}$ satisfying (10) for some subsequence $\{k_1, k_2, \dots\} \subset K$. Let $\widetilde{z} = \lim_{q\to\infty} \widetilde{z}^{k_q}$ and assume, to the contrary, that $\widetilde{z} > f(\mathbf{x}')$ for some $\mathbf{x}' \in D$. For every $q = 1, 2, \dots$, this point $\mathbf{x}'$ belongs to some $\Delta \in \mathcal{L}$ and we have

$$\widetilde{z}^{k_q} > f(\mathbf{x}') \geq w_R(\Delta) \geq w_R(\Delta^{k_q}),$$

because Δ^{k_q} is selected as a simplex with the least value of w_R at iteration k_q. However, $\widetilde{z}^{k_q} - w_R(\Delta^{k_q}) \to 0$ as $q \to +\infty$, which is a contradiction. Therefore, we have

$$\widetilde{z} \leq f(\mathbf{x}), \quad \forall \mathbf{x} \in D.$$

Let $\widetilde{\mathbf{x}}$ be any accumulation point of $\{\widetilde{\mathbf{x}}^k \mid k = 1, 2, \dots\}$ and assume that $\{\widetilde{\mathbf{x}}^{k_r} \mid r = 1, 2, \dots\}$ converges to $\widetilde{\mathbf{x}}$. Even in this subsequence, as $r \to +\infty$, we have

$$f(\widetilde{\mathbf{x}}^{k_r}) = \widetilde{z}^{k_r} \to \widetilde{z},$$

because $\{\widetilde{z}^k \mid k = 1, 2, \dots\}$ is nonincreasing. Hence, $\widetilde{\mathbf{x}}$ is a globally optimal solution to (1). □

Corollary 4.3. *If $K < +\infty$ or $\epsilon > 0$, the algorithm* simplicial_bb *terminates after finitely many iterations. Especially when $K = +\infty$ and $\epsilon > 0$, it yields a globally ϵ-optimal solution $\widetilde{\mathbf{x}}^k$ of (1).*

4.2 Application as heuristic tools

In order to use simplicial_bb to solve (1) heuristically, we need only to set K to a finite number. As will be seen in the next section, the algorithm requires few iterations and yields a feasible solution of almost the same value as an optimal solution in most cases. Even for a rather small K, we could expect feasible solutions of enough accuracy in actual applications. This is mainly due to the fact that local_search works well after solving R(Δ) as a heuristic tool. In fact, the algorithm simplicial_bb can be thought of as a kind of multistart algorithm where the starting points are generated systematically by branch-and-bound. The centroids $\mathbf{u}^k$'s of all simplices $\Delta^k \in \mathcal{L}$ serve as those starting points.

If we select $\mathbf{u}^k$'s randomly from the initial simplex Δ^1, as in the genuine multistart algorithm, we can obtain a complete heuristic algorithm for (1). It is described simply as follows:

```
algorithm simplicial_ms
begin
    compute an $n$-simplex $\Delta^1 = \mathrm{conv}(\{\mathbf{v}_1^1, \dots, \mathbf{v}_{n+1}^1\})$ including $D$;
    $\delta := 0$;
    for $k = 1, \dots, K$ do begin
        select a point $\mathbf{u}$ randomly from $\Delta^1$ and let $h(\mathbf{x}) := \nabla f(\mathbf{u})\mathbf{x}$;
        solve $\mathrm{R}(\Delta^1)$ of minimizing $h(\mathbf{x})$ and obtain its optimal basic solution $\mathbf{x}^k$;
        $\widetilde{\mathbf{x}}^k :=$ local_search$(\mathbf{x}^k)$
    end;
    select $\widetilde{\mathbf{x}} \in \arg\min\{f(\mathbf{x}) \mid \mathbf{x} = \widetilde{\mathbf{x}}^1, \dots, \widetilde{\mathbf{x}}^K\}$ and let $\widetilde{z} := f(\widetilde{\mathbf{x}})$;
    output $(\widetilde{\mathbf{x}}, \widetilde{z})$
end;
```

Note in this description that we may ignore the constant δ in the objective function h of $\mathrm{R}(\Delta^1)$ because we do not need lower bounds of f any more. In other words, $\mathrm{R}(\Delta^1)$ is solved just to provide a feasible basic solution for the function local_search.

5 Numerical results

In this section, we will report some numerical results of having compared the algorithms simplicial_bb, simplicial_ms and an ordinary simplicial branch-and-bound algorithm with bisection, referred to as textbook_bb. The test problem was a concave quadratic minimization problem of the form:

$$\left| \begin{array}{ll} \text{minimize} & f(\mathbf{x}) + \omega \mathbf{d}_y \mathbf{y} \\ \text{subject to} & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} \leq \mathbf{b}, \quad (\mathbf{x}, \mathbf{y}) \geq \mathbf{0}, \end{array} \right. \tag{12}$$

where

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^{\mathsf{T}}\mathbf{C}\mathbf{x} + \mathbf{d}_x\mathbf{x},$$

and $\mathbf{A} \in \mathbb{R}^{m \times p}$, $\mathbf{B} \in \mathbb{R}^{m \times q}$, $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{C} \in \mathbb{R}^{p \times p}$, $\mathbf{d}_x^{\mathsf{T}} \in \mathbb{R}^p$, $\mathbf{d}_y^{\mathsf{T}} \in \mathbb{R}^q$, and ω is a positive weight. The total number of variables is $n = p + q + m$, including m slack variables. The matrix $\mathbf{C} = [c_{ij}]$ was symmetric and tridiagonal, i.e., $c_{11} = c_{ii} = -1.0$, and $c_{i,i-1} = c_{i-1,i}$ was a uniformly random number in the interval $[-1.0, 0.0]$ for $i = 2, \dots, p$. Also, each component of $(\mathbf{d}_x, \mathbf{d}_y)$ was drawn randomly from the uniform distribution on $[-1, 0, 0.0]$. To make the feasible set bounded, $\mathbf{b}$ was set to $[1, \dots, 1, p + q]^{\mathsf{T}}$ and all components in

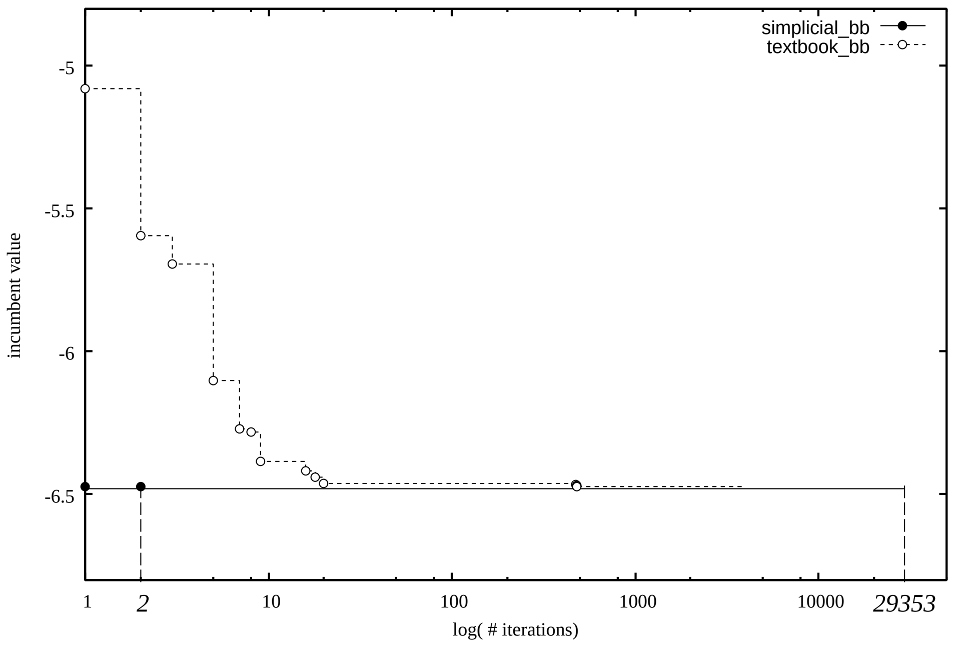

Figure 2: Changes in the incumbent value in the proposed algorithm.

the last row of $[\mathbf{A}, \mathbf{B}]$ were fixed at ones. Other components were all random numbers in the interval $[-0.5, 1.0]$, where the percentages of zeros and negative numbers were about 20% and 10%, respectively.

Computer codes of simplicial_bb, simplicial_ms and textbook_bb were written using GNU Octave (version 2.1.73), a MATLAB-like computational tool [15], and run on a single-processor workstation (Athlon64, 2.4GHz). Notice on the problem (12) that the nonlinear function f depends only on $\mathbf{x} \in \mathbb{R}^p$. Therefore, using the decomposition technique [8, 18], we defined the initial simplex Δ^1 so as to include the projection of the feasible set onto the space $\mathbb{R}^p$ of $\mathbf{x}$, and carried out essential operations related to subsimplices Δ's in $\mathbb{R}^p$, in each code. As for the optimality criterion in simplicial_bb and textbook_bb, we adopted $\tilde{z}^k - w_R(\Delta^k) \leq \epsilon|\tilde{z}^k|$, instead of $\tilde{z}^k - w_R(\Delta^k) \leq \epsilon$, to prevent the magnitude of $\tilde{z}^k$ from affecting the convergence, where ϵ was fixed at 10^{-5}. Incidentally, the example in Section 1 is an instance of (12) with $(m, n, p, \omega) = (40, 100, 20, 3.0)$, and if simplicial_bb is applied to it, the incumbent value changes as in Figure 2.

First, let us see the performance of simplicial_ms as an exact algorithm on small-scale instances with $(m, n) = (40, 120)$. Figures 3 and 4 show the variation in the average number of iterations and average computational time in seconds, respectively, required by simplicial_bb and textbook_bb to solve ten instances when p was fixed at 16 and ω ranged from 1.0 to 10.0. Figures 5 and 6 show it when ω was fixed at 3.0 and p ranged from 8 to 22. We see from these figures that simplicial_bb takes considerably more iterations than textbook_bb while there is not so much difference in the computational

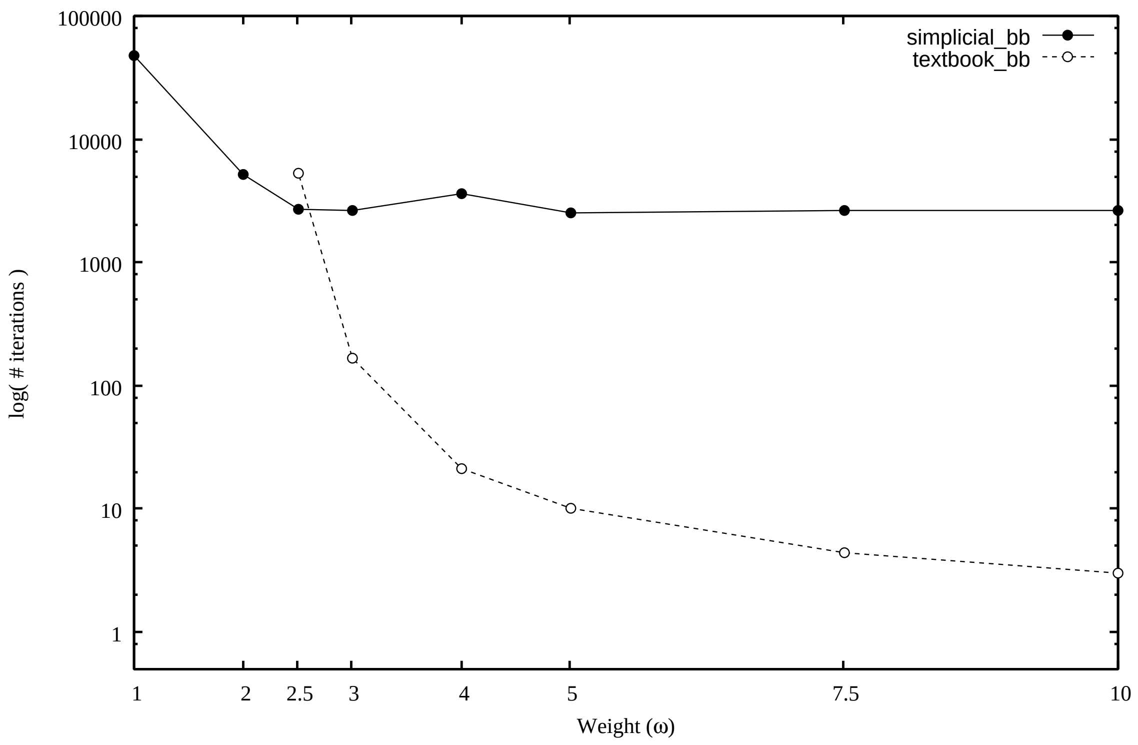

Figure 3: The numbers of iterations when $(m, n, p) = (40, 120, 16)$.

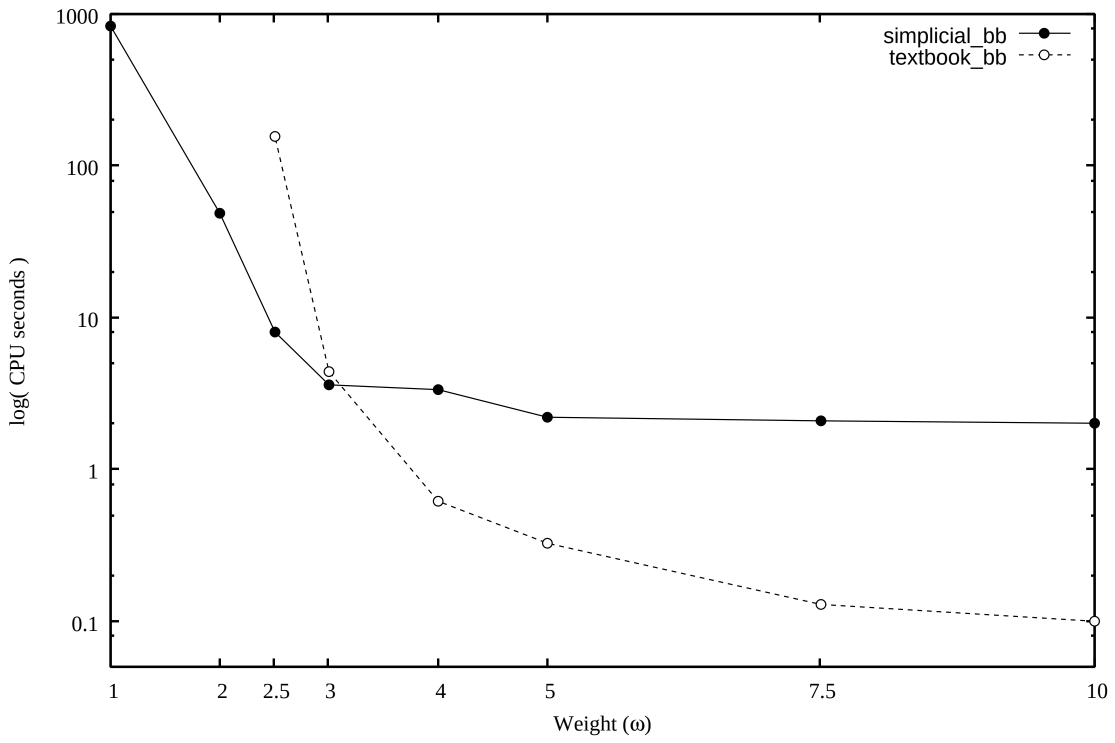

Figure 4: Computational time (seconds) when $(m, n, p) = (40, 120, 16)$.

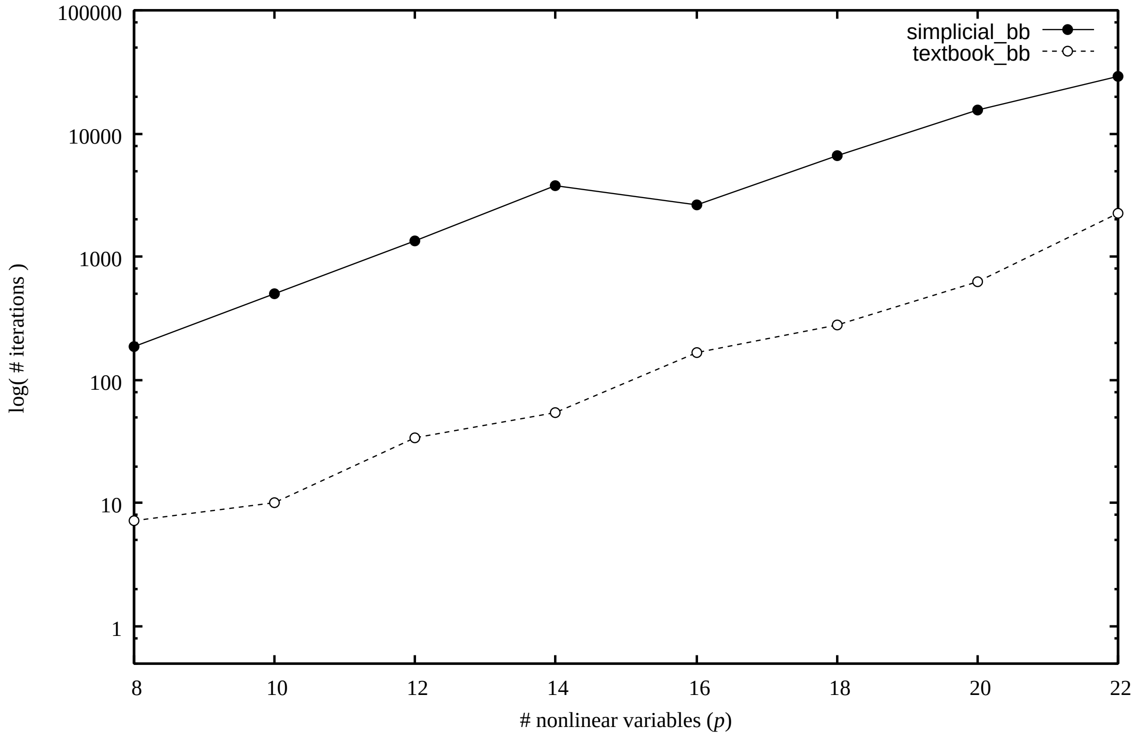

Figure 5: The numbers of iterations when $(m, n, \omega) = (40, 120, 3.0)$.

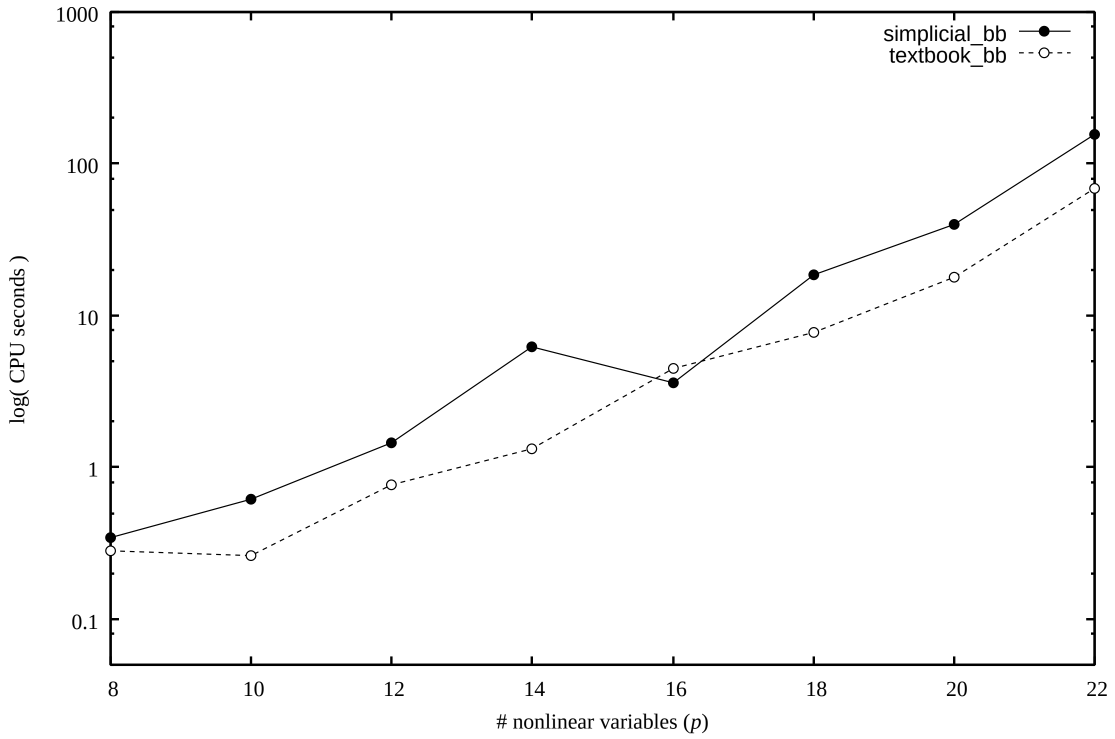

Figure 6: Computational time (seconds) when $(m, n, \omega) = (40, 120, 3.0)$.

Table 1: Performance for 32 iterations of simplicial_bb and simplicial_ms.

				$p = n/6$		$p = n/3$		$p = n/2$	
m	n	ω		*pivot*	*time*	*pivot*	*time*	*pivot*	*time*
100	300	1.0	bb	4810.9	5.1314	5923.5	6.293	5151.7	5.736
			ms	4877.3	4.471	6083.8	5.552	5534.6	5.320
200	600	1.0	bb	7995.1	30.23	11677	45.09	15906	63.73
			ms	8126.2	28.52	12115	43.58	15157	56.55
200	600	3.0	bb	3326.1	12.44	4356.4	16.33	5219.0	19.83
			ms	3254.1	11.42	4422.2	15.42	5207.0	18.17
300	900	1.0	bb	14408	115.4	21634	173.5	27147	222.9
			ms	15216	115.3	21379	164.7	27045	214.6
300	900	3.0	bb	7937.4	63.34	10962	87.60	13027	104.6
			ms	8335.4	63.89	11055	83.41	12895	97.38
400	1,200	3.0	bb	15800	209.2	20224	267.7	22856	302.5
			ms	15171	191.4	20484	258.8	22842	288.8

time. This dominance relation is reversed when ω is less than about 3.0. In fact, the average computational time of textbook_bb was well over $2,000$ seconds for problem instances with $\omega \leq 2.0$. Once ω is fixed, however, both codes behaves alike, as shown in Figure 6. As a whole, the algorithm simplicial_bb appears to have potential for efficiency, at least comparable to the usual simplicial branch-and-bound algorithm.

Let us turn next to the performance of simplicial_bb as a heuristic algorithm. We fixed the number K of iterations at 32 in simplicial_bb, simplicial_ms, and solved instances of size up to $(m, n, p) = (400, 1200, 600)$, using both codes. Table 1 lists the average number of simplex pivots (*pivot*) and average computational time in seconds (*time*) required by simplicial_bb (bb) and simplicial_ms (ms) to solve ten instances. Both codes carry out almost the same number of pivoting operations, but simplicial_bb takes a little more computational time because of additional operations involved in branch-and-bound. To investigate the quality of solutions, we extract some outputs (for instances #1 and #2) of both codes and list them in Table 2, which includes the incumbent value (*output*) and lower bound (*bound*) at termination, and the iteration number at which the incumbent was last updated (*update*). For problem instances of size $(m, n, p, w) = (200, 600, 100, 3.0)$ and $(300, 900, 150, 3.0)$, we also add the optimal values, which were computed using a code of the algorithm developed in [9]. We observed for every instance of these two sizes that both simplicial_bb and simplicial_ms generated a feasible solution of the same value as an optimal solution. Unfortunately, each instance of the other sizes is too large to solve to optimality using our available computer codes. However, simplicial_bb always

Table 2: Outputs after 32 iterations of simplicial_bb and simplicial_ms.

m	*n*	*p*	*ω*		*opt.val.*		*output*	*bound*	*update*
200	600	100	3.0	#1	-7.441762	bb	-7.441762	-11.550959	1
						ms	-7.441762	—	31
				#2	-7.844702	bb	-7.844702	-11.819086	1
						ms	-7.844702	—	1
300	900	150	3.0	#1	-7.708254	bb	-7.708254	-11.948366	1
						ms	-7.708254	—	1
				#2	-7.689507	bb	-7.689507	-11.776019	1
						ms	-7.689507	—	1
300	900	450	1.0	#1	—	bb	-2.704720	-7.620925	21
						ms	-2.704726	—	10
				#2	—	bb	-2.681544	-7.643540	27
						ms	-2.681603	—	25
300	900	450	3.0	#1	—	bb	-6.355546	-11.316951	1
						ms	-6.355546	—	1
				#2	—	bb	-6.567183	-11.560518	2
						ms	-6.567183	—	13
400	1,200	600	3.0	#1	—	bb	-6.996379	-11.794577	1
						ms	-6.996379	—	1
				#2	—	bb	-6.982920	-11.796606	1
						ms	-6.982920	—	1

maintains the lower bound, which provides an indication, though rough, of the accuracy of outputs. Namely, we see from Table 2 that the ratio of the output value of simplicial_bb to the optimal value is greater than 0.35 when $\omega = 1.0$ and it rises to greater than 0.6 when $\omega = 3.0$. This is an important strength of simplicial_bb, not shared by other heuristic algorithms.

6 Conclusion

To enhance the computational efficiency of the branch-and-bound algorithm, various techniques of tightening the lower bound have been developed so far. In contrast to those, we have loosened it, and instead tightened the upper bound in this paper. As a result, we have shown that good feasible solutions are obtained in the quite early stage of iterations and the algorithm can be used as a heuristic tool as well. The algorithm simplicial_bb is our answer to the present situation where none can ask for theoretically efficient algorithms for the concave minimization problem. However, it is not a final one because there remain two substantive issues to be improved in simplicial_bb. First, its computational efficiency still pales in comparison with other exact algorithms. To put simplicial_bb into practical use as an exact algorithm, we need to incorporate some procedures of tightening the lower bound, such as a Lagrangian-based one in [9, 10, 12]. The other issue is memory consumption of the implementation. Since simplicial_bb stores an exponential number of Δ^k's in the worst case, it may suffer from a combinatorial explosion in memory consumption, before yielding a solution, if we try to solve a large scale instance to optimality. This is a critical issue shared by many branch-and-bound algorithms for the concave minimization problem. The reverse search technique in [11] could be a hint towards a possible resolution.

References

[1] Chvátal, V., *Linear Programming*, Freeman (N.Y., 1983).

[2] Karmarkar, N., "A new polynomial-time algorithm for linear programming", *Combinatorica* **4** (1984), 373–395.

[3] McMullen, P., "The maximum numbers of faces of a convex polytope", *Mathematika* **17** (1970), 179–184.

[4] Horst, R., "An algorithm for nonconvex programming problems", *Mathematical Programming* **10** (1976), 312–321.

[5] Horst, R., P.M. Pardalos, and N.V. Thoai, *Introduction to Global Optimization*, Springer-Verlag (Berlin, 1995).

[6] Guisewite, G.M., and P.M. Pardalos, "Minimum concave-cost network flow problems: applications, complexity, and algorithms", *Annals of Operations Research* **25** (1990), 75–100.

[7] Holmberg, K., and H. Tuy, "A production-transportation problem with stochastic demand and concave production costs", *Mathematical Programming* **85** (1999), 157–179.

[8] Horst, R. and H. Tuy, *Global Optimization: Deterministic Approaches*, 2nd ed., Springer-Verlag (Berlin, 1993).

[9] Kuno, T., and H. Nagai, "A simplicial algorithm with two-phase bounding operation for a class of concave minimization problems", *Pacific Journal of Optimization* **1** (2005), 277–296.

[10] Kuno, T., and H. Nagai, "A simplicial branch-and-bound algorithm conscious of special structures in concave minimization problems", *CS Technical Report* **05-2** (University of Tsukuba, 2005), to appear in *Computational Optimization and Applications*.

[11] Kuno, T., and Y. Shiguro, "A polynomial-space finite algorithm for solving a class of reverse convex programs", *CS Technical Report* **07-8** (University of Tsukuba, 2007).

[12] Kuno, T., and T. Utsunomiya, "A Lagrangian based branch-and-bound algorithm for production-transportation problems", *Journal of Global Optimization* **18** (2000), 59–73.

[13] Locatelli, M., and U. Raber, "On convergence of the simplicial branch-and-bound algorithm based on ω-subdivisions", *Journal of Optimization Theory and Applications* **107** (2000), 69–79.

[14] Nagai, H., and T. Kuno, "A simplicial branch-and-bound algorithm for production-transportation problems with inseparable concave production cost", *Journal of the Operations Research Society of Japan* **48** (2005), 90–110.

[15] Octave Home Page, http://www.octave.org/.

[16] Vavasis, S.A., *Nonlinear Optimization: Complexity Issues*, Oxford University Press (Oxford, 1991).

[17] Tuy, H., "Concave programming under linear constraints", *Soviet Mathematics* **5** (1964), 1437–1440.

[18] Tuy, H., *Convex Analysis and Global Optimization*, Kluwer Academic Publishers (Dordrecht, 1998).