A modified simplicial algorithm for convex maximization based on an extension of ω-subdivision

Takahito Kuno

Graduate School of Systems and Information Engineering

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

March, 2017
Revised: January 30, 2018

Abstract

The simplicial algorithm is a popular branch-and-bound approach to the convex maximization problem with multiple local maxima. In this paper, we discuss some difficulties revealed when implementing this algorithm under the ω-subdivision rule. To overcome those, we modify the bounding process and extend the ω-subdivision rule. We also report numerical results for the simplicial algorithm according to the new subdivision rule.

Key words: Global optimization, convex maximization, branch-and-bound, simplicial algorithm, ω-subdivision.

1 Introduction

The convex maximization problem is hard to solve due to the existence of multiple local maxima, and known to be NP-hard from the viewpoint of computational complexity [5, 16]. To solve the problem to global optimality, we need to enumerate all local maxima either explicitly or implicitly [7, 8, 13, 18]. A typical approach to doing this is branch-and-bound, which locates a globally optimal solution by alternating two processes: branching and bounding. While the former partitions a simple polyhedron enclosing the feasible set into a number of subpolyhedra, the latter filters subproblems associated with those subpolyhedra by comparing their upper bounds with a lower bound given by some feasible solution found so far. According to the shape of the polyhedron used in these processes, branch-and-bound algorithms are classified into three classes: the conical algorithm [17], the simplicial algorithm [6], and the

The author was partly supported by a Grant-in-Aid for Scientific Research (C) 16K0028 from Japan Society for the Promotion of Science. E-mail: takahito@cs.tsukuba.ac.jp rectangular algorithm [3]. Of particular interest in this paper is the simplicial algorithm which was proposed by Horst in 1976.

In the bounding process of the simplicial algorithm, the objective function is approximated on a given simplex S into an affine function, which is maximized to obtain an upper bound for the associated subproblem. Therefore, we need to solve a linear programming problem in every execution of the bounding process. To reduce the time taken for this process, usually, the linearized subproblem is solved by the simplex algorithm, starting from the optimal solution of a problem solved in the preceding execution of the bounding process. If the difference between the current and preceding problems is small, this reoptimization requires little effort. However, for example, when the optimal solution of the preceding problem is infeasible for the current problem, it can requires unexpectedly many simplex pivots to recover the optimality for the current problem.

In this paper, we show that the computational efficiency of the simplicial algorithm is considerably improved by dropping the constraint on the simplex S from the linearized subproblem. As a consequence of this modification, the optimal solution $\boldsymbol{\omega}$ of the linearized subproblem does not always lie in S. In that case, we cannot apply the ω-subdivision rule in the branching process, despite its empirical efficiency [9, 10], since the rule is configured to partition S radially around $\boldsymbol{\omega}$ assumed to be in S. To cope with such a case, we develop a new simplicial subdivision rule, named extended ω-subdivision, which uses $\boldsymbol{\omega}$ to partition S even if $\boldsymbol{\omega}$ is not a point in S, and show that the simplicial algorithm works properly under this subdivision rule. In Section 2, we give a formal definition of the convex maximization problem, and illustrate how the simplicial algorithm behaves on the problem. In Section 3, we modify the linearized subproblem, and discuss possible advantages and disadvantages of using the resulting problem in the algorithm. In Section 4, to overcome the major disadvantage mentioned above, we develop the extended ω-subdivision rule, and examine the convergence properties of the algorithm according to it. In Section 5, after summarizing the algorithm incorporating the new subdivision rule, we report some numerical results to compare it with the usual simplicial algorithm. Lastly, we conclude the paper with some remarks in Section 6.

2 Convex maximization and the simplicial algorithm

Let f be a convex function defined on $\mathbb{R}^n$, and consider a problem of maximizing it on a polyhedron:

$$\left| \begin{array}{ll} \text{maximize} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{A}\mathbf{x} \le \mathbf{b}, \end{array} \right. \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{b} \in \mathbb{R}^m$. Let us denote the feasible set by

$$D = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \le \mathbf{b}\}, \tag{2}$$

and assume that D is nonempty and bounded. We also assume that an n-simplex $S^1 \subset \mathbb{R}^n$ is given as the convex hull of $n+1$ affinely independent points $\mathbf{v}_1^1, \dots, \mathbf{v}_{n+1}^1$ and satisfies

$$D \subset S^1 \subset \operatorname{int}(\operatorname{dom} f), \tag{3}$$

where dom $\cdot$ and int $\cdot$ represent the effective domain and the interior, respectively. Under these assumptions, the problem (1) has at least one optimal solution. However, there in general coexist multiple locally optimal solutions, most of which are not globally optimal. To obtain a globally optimal solution, we need to enumerate all locally optimal solutions, either explicitly or implicitly, using techniques such as branch-and-bound. One of standard branch-and-bound approaches is the simplicial algorithm, which is outlined below.

Outline of the simplicial algorithm

As in other branch-and-bound algorithms, the following two processes play essential roles in the simplicial algorithm.

Branching: Starting from $i = 1$, the n-simplex $S^i = \operatorname{conv}\{\mathbf{v}_1^i, \dots, \mathbf{v}_{n+1}^i\}$ is subdivided radially around a point $\mathbf{u} \in S^i$ into at most $n+1$ children:

$$S_j^i = \operatorname{conv}\{\mathbf{v}_1^i, \dots, \mathbf{v}_{j-1}^i, \mathbf{u}, \mathbf{v}_{j+1}^i, \dots, \mathbf{v}_{n+1}^i\}, \quad j \in J^i, \tag{4}$$

where J^i is an index set such that $j \in J^i$ if $\mathbf{v}_1, \dots, \mathbf{v}_{j-1}^i, \mathbf{u}, \mathbf{v}_{j+1}^i, \dots, \mathbf{v}_{n+1}^i$ are affinely independent. We refer to $\mathbf{u}$ as the *subdivision point* of S^i. Out of active descendants of S^1 requiring further examination, a simplex is chosen as the successor S^{i+1} to S^i, and the same process is repeated after incrementing i by one, until all active descendants turn out to contain no optimal solution of (1).

Bounding: Except in the trivial case where $D \cap S^i = \emptyset$, whether or not S^i needs to be subdivided is determined by comparing an upper bound $\beta(S^i)$ of f on $D \cap S^i$ with the value α of f at the best known feasible solution. The value of $\beta(S^i)$ is given by maximizing the concave envelope g^i of f, the pointwise infimum over all concave overestimators of f on S^i. In our case where f is a convex function, g^i is an affine function which agrees with f at the vertices of S^i. Therefore, a maximum point $\boldsymbol{\omega}^i$ of g over $D \cap S^i$ can be obtained by linear programming. If $\alpha \geq \beta(S^i)$ holds for $\beta(S^i) = g^i(\boldsymbol{\omega}^i)$, then S^i contains no feasible solution of value better than α and can be pruned from the set of active descendants of S^1.

If the algorithm does not terminate in a finite amount of time, it generates an infinite sequence of nested simplices:

$$S^1 = S^{i_1} \supset \cdots \supset S^{i_k} \supset S^{i_{k+1}} \supset \cdots,$$

where $S^{i_{k+1}}$ is a child of S^{i_k} created by subdividing S^{i_k} around some $\mathbf{u} \in S^{i_k}$. The convergence of the algorithm depends largely on how to subdivide S^i in the branching process. If the subdivision point $\mathbf{u}$ is placed at the midpoint on a longest edge of S^i for each i, then S^{i_k} shrinks to a single point as $k \to \infty$. Since $\boldsymbol{\omega}^{i_k}$ belongs to S^{i_k}, we simultaneously have

$$\liminf_{k \to \infty} \left(g^{i_k}(\boldsymbol{\omega}^{i_k}) - f(\boldsymbol{\omega}^{i_k}) \right) = 0.$$

This guarantees the convergence of the algorithm to an optimal solution of (1) if the successor S^{i+1} to S^i is chosen in *best-bound-first order*, i.e., S^{i+1} is a simplex with the largest upper bound among all active descendants of S^1. In addition to this simple *bisection*, there are several rules for subdividing S^i which guarantee the convergence of the algorithm [9, 10, 11, 12]. Among others, the most poplar is the ω-*subdivision* rule, where $\mathbf{u}$ is placed at $\boldsymbol{\omega}^i$ for each i. Empirically, it is known that the ω-subdivision rule runs the algorithm much more efficiently than bisection [9]. Whichever rule is adopted, in order to make the algorithm converge to an optimal solution of (1), the successor S^{i+1} to S^i has to be chosen in best-bound-first order.

3 Reduction of effort in the bounding process

As seen in the previous section, for a given n-simplex $S = \mathrm{conv}\{\mathbf{v}_1, \ldots, \mathbf{v}_{n+1}\} \subset S^1$, the bounding process needs an upper bound $\beta(S)$ for the subproblem of (1):

$$\left| \begin{array}{ll} \text{maximize} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{x} \in D \cap S. \end{array} \right. \tag{5}$$

Assume that $D \cap S \neq \emptyset$. If not, $\beta(S)$ may be set to $-\infty$. Replacing f with its concave envelope g on S, the subproblem (5) is approximated into a linearized problem:

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

Although $\mathrm{P}(S)$ is certainly linear programming in $\mathbb{R}^n$, to obtain its standard form, we have to explicitly determine the function g and the constraint $\mathbf{x} \in S$. Instead, to avoid those complications, the following equivalent problem in $\mathbb{R}^{n+1}$ is commonly solved:

$$\Pi(S) \left| \begin{array}{ll} \text{maximize} & \mathbf{f}\boldsymbol{\lambda} \\ \text{subject to} & \mathbf{A}\mathbf{V}\boldsymbol{\lambda} \leq \mathbf{b}, \quad \mathbf{e}\boldsymbol{\lambda} = 1, \quad \boldsymbol{\lambda} \geq 0, \end{array} \right.$$

where $\mathbf{e} \in \mathbb{R}^{n+1}$ is the all-ones row vector, and

$$\mathbf{f} = [f(\mathbf{v}_1), \dots, f(\mathbf{v}_{n+1})], \quad \mathbf{V} = [\mathbf{v}_1, \dots, \mathbf{v}_{n+1}].$$

Let $\boldsymbol{\lambda}^*$ be an optimal solution to $\Pi(S)$. Then $\boldsymbol{\omega} = \mathbf{V}\boldsymbol{\lambda}^*$ solves $\mathrm{P}(S)$, and $\beta(S)$ is given by the optimal value $g(\boldsymbol{\omega}) = \mathbf{f}\boldsymbol{\lambda}^*$ of $\mathrm{P}(S)$ and $\Pi(S)$.

In the usual implementation of the simplicial algorithm, when S^{i+1} is given as the successor to the current simplex S^i, problem $\Pi(S^{i+1})$ is solved by performing a sequence of dual and primal simplex pivots from an optimal solution of $\Pi(S^i)$. If S^{i+1} is a child of S^i, then $\Pi(S^{i+1})$ differs from $\Pi(S^i)$ only in a column of the profit vector and constraint matrix, and so this reoptimization needs few simplex pivots (see e.g., [2]). If not, however, it can require a large number of simplex pivots because $\Pi(S^i)$ and $\Pi(S^{i+1})$ are significantly different. An easy way to reduce this effort is to choose the successor S^{i+1} in *depth-first order*. Then S^{i+1} is always a child of S^i except when S^i is pruned from the set of active descendants of S^1. While this approach does not guarantee the convergence to a globally optimal solution of (1), for any given tolerance $\varepsilon > 0$, the algorithm still terminates in finite time and returns a globally ε-optimal solution $\mathbf{x}^\varepsilon \in D$ which satisfies

$$f(\mathbf{x}^\varepsilon) \geq f(\mathbf{x}) - \varepsilon, \quad \forall \mathbf{x} \in D.$$

In the following, we will conduct a more drastic revision of the simplicial algorithm.

Alternative linearized subproblem

As an alternative to $\Pi(S)$, we propose to solve the following in the bounding process:

$$\mathrm{P}'(S) \left| \begin{array}{ll} \text{maximize} & \mathbf{c}'\mathbf{x} + c'_0 \\ \text{subject to} & \mathbf{A}\mathbf{x} \leq \mathbf{b}. \end{array} \right.$$

The profit vector $[\mathbf{c}', c'_0]$ is a solution to the system of linear equations:

$$[\mathbf{c}, c_0] \begin{bmatrix} \mathbf{V} \\ \mathbf{e} \end{bmatrix} = \mathbf{f}. \tag{6}$$

Therefore, the objective function of $\mathrm{P}'(S)$ is just the concave envelope of f over S. Since $D = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \leq \mathbf{b}\}$ is assumed to be nonempty, $\mathrm{P}'(S)$ always has an optimal solution $\boldsymbol{\omega}'$ in D, for which we have

$$g(\boldsymbol{\omega}') = \mathbf{c}'\boldsymbol{\omega}' + c'_0 \geq g(\boldsymbol{\omega}) \geq f(\mathbf{x}), \quad \forall \mathbf{x} \in D \cap S. \tag{7}$$

Therefore, $g(\boldsymbol{\omega}')$ can be used as the upper bound $\beta(S)$ for the subproblem (5). Furthermore, whichever descendant of S^1 is chosen as the successor S^{i+1} to S^i, problem $\mathrm{P}'(S^{i+1})$ differs from $\mathrm{P}'(S^i)$ only in the profit vector, and can be reoptimized with only a few primal simplex pivots. The substitution of $\mathrm{P}'(S)$ for $\Pi(S)$, however, has three obvious disadvantages:

(a) the upper bound $\beta(S)$ deteriorates in quality, as shown in (7);

(b) an additional effort is needed to solve the system (6); and

(c) the solution $\boldsymbol{\omega}'$ cannot be the subdivision point of S if $\boldsymbol{\omega}' \notin S$.

Among these weak points, (b) can be minor if the successor S^{i+1} to S^i is chosen in depth-first order. Let $S^i = \mathrm{conv}\{\mathbf{v}_1^i, \ldots, \mathbf{v}_{n+1}^i\}$ and $\mathbf{V}^i = [\mathbf{v}_1^i, \ldots, \mathbf{v}_{n+1}^i]$. Unless S^i is pruned off, the successor S^{i+1} is a child of S^i, and hence the associated matrix $\mathbf{V}^{i+1}$ differs from $\mathbf{V}^i$ only in one column, say $\mathbf{v}_j^{i+1} = \mathbf{u}$, which is the subdivision point of S^i. We can update the inverse of the coefficient matrix of (6) in time $O(n^2)$, using the Sherman-Morrison-Woodbury formula (see for detail, e.g., [14]), as follows

$$\mathbf{W}^{i+1} = \left(\mathbf{I} - \frac{1}{\mathbf{e}_j\mathbf{z}}\left(\mathbf{z} - \mathbf{e}_j^\top\right)\mathbf{e}_j\right)\mathbf{W}^i, \tag{8}$$

where $\mathbf{I} \in \mathbb{R}^{(n+1)\times(n+1)}$ is the identity matrix, $\mathbf{e}_j \in \mathbb{R}^{n+1}$ is its jth row, and

$$\mathbf{W}^i = \begin{bmatrix} \mathbf{V}^i \\ \mathbf{e} \end{bmatrix}^{-1}, \quad \mathbf{z} = \mathbf{W}^i \begin{bmatrix} \mathbf{u} \\ 1 \end{bmatrix}.$$

Since the usual reoptimization procedure performs a single simplex pivot in time $O(m^2)$ using a formula similar to (8), the computational burden for solving (6) would be offset by solving $\mathrm{P}'(S)$ instead of $\Pi(S)$, as long as n is not extremely larger than m.

Subdivision of the simplex S using $\boldsymbol{\omega}'$

The weakness (c) in the use of $\mathrm{P}'(S)$ is naturally attributed to dropping the constraint on S from $\mathrm{P}(S)$. Let us develop a way to exploit the optimal solution $\boldsymbol{\omega}'$ of $\mathrm{P}'(S)$ to subdivide S even in the case where $\boldsymbol{\omega}'$ is not a point in S.

Once the system (6) has been solved, the following can be solved in $\boldsymbol{\mu}$ with a little additional effort:

$$\begin{bmatrix} \mathbf{V} \\ \mathbf{e} \end{bmatrix} \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\omega}' \\ 1 \end{bmatrix}. \tag{9}$$

Let $\boldsymbol{\mu}'$ denote the solution to (9), and let $J_+ = \{j \mid \mu'_j > 0\}$. Then, by letting

$$\lambda'_j = \begin{cases} \mu'_j / \sum_{k \in J_+} \mu'_k & \text{if } j \in J_+ \\ 0 & \text{otherwise,} \end{cases} \tag{10}$$

we have $\boldsymbol{\lambda}' \geq 0$ and $\mathbf{e}\boldsymbol{\lambda}' = 1$. Therefore, if we define $\mathbf{u}$ as follows, then it belongs to S without fail, and can serve as the subdivision point of S:

$$\mathbf{u} = \mathbf{V}\boldsymbol{\lambda}'. \tag{11}$$

Since $\mathbf{e}\boldsymbol{\mu}' = 1$ holds, J_+ never vanishes, and hence $\mathbf{u}$ is well-defined through (9) – (11). However, if $\mathbf{u}$ falls in the vertex set $\{\mathbf{v}_1, \ldots, \mathbf{v}_{n+1}\}$ of S, then S cannot be subdivided around $\mathbf{u}$. In that case, we can prune S from the set of active descendants of S^1.

Proposition 3.1 *Let* $\mathbf{u}$ *be defined through (9) – (11). If* $\mathbf{u} \in \{\mathbf{v}_1, \ldots, \mathbf{v}_{n+1}\}$*, then* $f(\boldsymbol{\omega}') \geq f(\mathbf{x})$ *for any* $\mathbf{x} \in D \cap S$*.*

Proof. Suppose $\mathbf{u} = \mathbf{v}_k$. If $\lambda'_k < 1$, then we have

$$\mathbf{v}_k = \frac{1}{1-\lambda'_k} \sum_{j \neq k} \lambda'_j \mathbf{v}_j, \quad \frac{1}{1-\lambda'_k} \sum_{j \neq k} \lambda'_j = 1, \quad \frac{1}{1-\lambda'_k} \lambda'_j \geq 0, \quad j \neq k,$$

which imply $\mathbf{v}_k \in \operatorname{conv}\{\mathbf{v}_j \mid j \neq k\}$. This contradicts the fact that S is an n-simplex. Therefore, $\lambda'_k = 1$, $J_+ = \{k\}$, and we have $\mu'_k \geq 1$ and $\mu'_j \leq 0$ for every $j \neq k$.

If $\mu'_k = 1$, then $\boldsymbol{\omega}' = \mathbf{v}_k$, and hence $f(\boldsymbol{\omega}') = g(\boldsymbol{\omega}') \geq g(\mathbf{x}) \geq f(\mathbf{x})$ for any $\mathbf{x} \in D \cap S$. Assume that $\mu'_k > 1$. Let

$$\mathbf{y} = \frac{1}{1-\mu'_k} \sum_{j \neq k} \mu'_j \mathbf{v}_j.$$

Then $\mathbf{y}$ belongs to a facet $\operatorname{conv}\{\mathbf{v}_j \mid j \neq k\}$ of S, since

$$\frac{1}{1-\mu'_k} \sum_{j \neq k} \mu'_j = 1, \quad \frac{1}{1-\mu'_k} \mu'_j \geq 0, \quad j \neq k.$$

Moreover, as is seen below, $\mathbf{v}_k$ lies in the relative interior of the segment connecting $\mathbf{y}$ and $\boldsymbol{\omega}'$:

$$\mathbf{v}_k = (1 - \frac{1}{\mu'_k})\mathbf{y} + \frac{1}{\mu'_k} \boldsymbol{\omega}', \quad \frac{1}{\mu'_k} \in (0, 1).$$

If $f(\boldsymbol{\omega}') < g(\boldsymbol{\omega}')$, we have

$$\begin{aligned} f(\mathbf{v}_k) = f\left((1-\frac{1}{\mu_k'})\mathbf{y}+\frac{1}{\mu_k'}\boldsymbol{\omega}'\right) &\leq (1-\frac{1}{\mu_k'})f(\mathbf{y})+\frac{1}{\mu_k'}f(\boldsymbol{\omega}') \\ &< (1-\frac{1}{\mu_k'})g(\mathbf{y})+\frac{1}{\mu_k'}g(\boldsymbol{\omega}') \\ &= g\left((1-\frac{1}{\mu_k'})\mathbf{y}+\frac{1}{\mu_k'}\boldsymbol{\omega}'\right) = g(\mathbf{v}_k), \end{aligned}$$

by noting $f(\mathbf{y}) \leq g(\mathbf{y})$. This is a contradiction, because g agrees with f at $\mathbf{v}_k$. As a consequence, we have $f(\boldsymbol{\omega}') \geq g(\boldsymbol{\omega}') \geq g(\mathbf{x}) \geq f(\mathbf{x})$ for any $\mathbf{x} \in D \cap S$. $\square$

4 New subdivision rule in the branching process

In the branching process, let us suppose that the successor S^{i+1} to S^i is chosen in depth-first order for each i. In that case, if the algorithm does not terminate, it generates an infinite sequence of nested simplices:

$$S^1 \supset S^2 \supset \cdots \supset S^i \supset S^{i+1} \supset \cdots, \tag{12}$$

where S^{i+1} is a child of S^i and shares n vertices with S^i. Let $\boldsymbol{\omega}^i$ denote the optimal solution of $\mathrm{P}'(S^i)$. The following relationship is assumed between the values of f and its concave envelope g^i on S^i:

$$g^i(\boldsymbol{\omega}^i) > f(\boldsymbol{\omega}^i), \quad i = 1,2,\ldots. \tag{13}$$

If the inequality does not hold for some i, we can terminate the generation of (12) because S^i and its successors contain no feasible solution of (1) better than $\boldsymbol{\omega}^i$. Let $S^0 = \cap_{i=1}^{\infty} S^i$. It is known that S^0 is a simplex but the dimension can be less than n [1].

If there is a large variability in the edge lengths of S^i, we cannot obtain stable solutions to the systems (6) and (9). A simple way to avoid this is to occasionally apply the usual bisection rule in (12) and shorten a longest edge of S^i by half. In general, such a hybrid subdivision rule with bisection enjoys some favorable property (see the textbook [18] for a proof):

Lemma 4.1 *For each i, let $\mathbf{x}^i$ be a point in S^i. For the sequence (12), assume that*

(i) for infinitely many i, simplex S^{i+1} is created by bisection of S^i,

(ii) for all other i, simplex S^{i+1} is created by subdividing S^i radially around $\mathbf{x}^i$.

Then at least one accumulation point of the sequence $\{\mathbf{x}^i\}$ is a vertex of S^0.

We also have the following, even without the assumptions (i) and (ii) in Lemma 4.1:

Lemma 4.2 *For each i, let $\mathbf{x}^i$ be a point in S^i. If at least one accumulation point of the sequence $\{\mathbf{x}^i\}$ is a vertex of S^0, then*

$$\liminf_{i\to\infty}\left(g^i(\mathbf{x}^i) - f(\mathbf{x}^i)\right) = 0. \tag{14}$$

Proof. Suppose a subsequence $\{\mathbf{x}^{i_r}\}$ converges to $\mathbf{x}^0$, which is a vertex of S^0. Let us abbreviate the index i_r to r. Since $\mathbf{x}^r \in S^r$, there exists some $\boldsymbol{\lambda}^r \geq \mathbf{0}$ such that $\mathbf{x}^r = \mathbf{V}^r\boldsymbol{\lambda}^r$ and $\mathbf{e}\boldsymbol{\lambda}^r = 1$. Passing to a further subsequence if necessary, we have $\mathbf{v}_j^r \to \mathbf{v}_j^0$ for each j and $\boldsymbol{\lambda}^r \to \boldsymbol{\lambda}^0$, as $r \to \infty$, for some $\mathbf{v}_j^0 \in S^1$ and $\boldsymbol{\lambda}^0 \geq \mathbf{0}$ such that $\mathbf{e}\boldsymbol{\lambda}^0 = 1$. Then $S^0 = \mathrm{conv}\{\mathbf{v}_j^0 \mid j = 1,\ldots,n+1\}$ holds (see [1]). Taking the limits of both sides of $\mathbf{x}^r = \mathbf{V}^r\boldsymbol{\lambda}^r$, we have $\mathbf{x}^0 = \mathbf{V}^0\boldsymbol{\lambda}^0$, where $\mathbf{V}^0 = [\mathbf{v}_1^0,\ldots,\mathbf{v}_{n+1}^0]$. Since $\mathbf{x}^0$ is a vertex of S^0, we see that $\mathbf{v}_j^0 = \mathbf{x}^0$ if $\lambda_j^0 > 0$. By the continuity of f, we have

$$g^r(\mathbf{x}^r) = \sum_{j=1}^{n+1} f(\mathbf{v}_j^r)\lambda_j^r \to \sum_{j=1}^{n+1} f(\mathbf{v}_j^0)\lambda_j^0 = f(\mathbf{x}^0) \quad (r \to \infty),$$

which implies (14) because $g^i(\mathbf{x}^i) \geq f(\mathbf{x}^i)$ for every i. $\square$

Let $\mathbf{u}^i$ denote the subdivision point S^i defined according to (9) – (11) with $\mathbf{V} = \mathbf{V}^i$ and $\boldsymbol{\omega}' = \boldsymbol{\omega}^i$. Since $\mathbf{u}^i \in S^i$ for each i, Lemmas 4.1 and 4.2 guarantee that there exists a subsequence $\{\mathbf{u}^{i_r}\}$ which converges to some vertex of S^0 and satisfies

$$\lim_{r\to\infty}\left(g^{i_r}(\mathbf{u}^{i_r}) - f(\mathbf{u}^{i_r})\right) = 0 \tag{15}$$

if the sequence (12) is generated, e.g., in accordance with the following rule for a prescribed positive integer N:

Extended ω-subdivision rule

(i) If $i = k \times N$ for some integer k, then S^i is bisected at the midpoint of a longest edge into two children, either of which is chosen as S^{i+1};

(ii) Otherwise, S^i is subdivided radially around $\mathbf{u}^i$ into at most $n+1$ children, one of which is chosen as S^{i+1}.

In addition to (15), we have a major consequence for the convergence of our algorithm under this subdivision rule.

Theorem 4.3 *If the sequence (12) is generated according to the extended ω-subdivision rule, then*

$$\liminf_{i\to\infty}\left(g^i(\boldsymbol{\omega}^i) - f(\boldsymbol{\omega}^i)\right) = 0. \tag{16}$$

Proof. Let $\{\mathbf{u}^{i_r}\}$ be a subsequence converging to $\mathbf{u}^0$, which is a vertex of S^0, and abbreviate i_r to r. If $\boldsymbol{\omega}^r = \mathbf{u}^r$ for infinitely many r, then (15) immediately implies (16). Suppose $\boldsymbol{\omega}^r = \mathbf{u}^r$ for only finitely many r. Let $\boldsymbol{\mu}^r$ denote the solution to (9) with $\mathbf{V} = \mathbf{V}^r$ and $\boldsymbol{\omega}' = \boldsymbol{\omega}^r$. By passing to a further subsequence if necessary, we can assume that there exists a subset $J_+ \subset \{1,\dots,n+1\}$ such that $J_+ = \{j \mid \mu_j^r > 0\}$ and $\sum_{j\in J_+} \mu_j^r > 1$ for every r. Let

$$\mathbf{y}^r = \frac{1}{\sum_{j\notin J_+} \mu_j^r} \sum_{j\notin J_+} \mu_j^r \mathbf{v}_j^r.$$

Then we have

$$\boldsymbol{\omega}^r = \mathbf{y}^r + \theta^r(\mathbf{u}^r - \mathbf{y}^r), \quad \theta^r = \sum_{j\in J_+} \mu_j^r > 1. \tag{17}$$

Since $\{\boldsymbol{\omega}^r\}$ and $\{\mathbf{y}^r\}$ are generated in compact sets, we can assume that they converge to some $\boldsymbol{\omega}^0 \in D$ and $\mathbf{y}^0 \in S^1$, respectively. There are two cases to consider.

Case 1: $\mathbf{u}^0 \neq \mathbf{y}^0$. If $\{\theta^r\}$ diverges, then $\{\boldsymbol{\omega}^r\}$ cannot be convergent. We can therefore assume that $\{\theta^r\}$ has a limit $\theta^0 \geq 1$. This simultaneously implies that $\{\boldsymbol{\mu}^r\}$ converges to some $\boldsymbol{\mu}^0$ such that $\mu_j^0 \geq 0$ if $j \in J_+$, and $\mu_j^0 \leq 0$ otherwise. First, let us assume that $\boldsymbol{\omega}^0 = \mathbf{u}^0$. Then θ^0 must be one, and $\mu_j^0 = 0$ for each $j \notin J_+$. As $r \to \infty$, we have

$$\begin{aligned} |g^r(\boldsymbol{\omega}^r) - f(\mathbf{u}^r)| &\leq |g^r(\boldsymbol{\omega}^r) - g^r(\mathbf{u}^r)| + |g^r(\mathbf{u}^r) - f(\mathbf{u}^r)| \\ &= |(1 - \frac{1}{\theta^r}) \sum_{j\in J_+} \mu_j^r f(\mathbf{v}_j^r) + \sum_{j\neq J_+} \mu_j^r f(\mathbf{v}_j^r)| + |g^r(\mathbf{u}^r) - f(\mathbf{u}^r)| \to 0, \end{aligned}$$

by noting (15), and hence $g^r(\boldsymbol{\omega}^r) \to f(\mathbf{u}^0) = f(\boldsymbol{\omega}^0)$. Next, assume $\boldsymbol{\omega}^0 \neq \mathbf{u}^0$ and that, contrary to (16), there exists some number $\sigma > 0$ such that $g^r(\boldsymbol{\omega}^r) - f(\boldsymbol{\omega}^r) > \sigma$ for sufficiently large r. Since f is convex and $g^r(\mathbf{y}^r) \geq f(\mathbf{y}^r)$, we have

$$\begin{aligned} f(\mathbf{u}^r) &\leq \frac{1}{\theta^r} f(\boldsymbol{\omega}^r) + (1 - \frac{1}{\theta^r}) f(\mathbf{y}^r) \\ &< \frac{1}{\theta^r}(g^r(\boldsymbol{\omega}^r) - \sigma) + (1 - \frac{1}{\theta^r}) g^r(\mathbf{y}^r) = g^r(\mathbf{u}^r) - \frac{\sigma}{\theta^r}, \end{aligned}$$

where the last equality holds in the view of the linearity of g^r. Thus, we have

$$\lim_{r\to\infty} (g^r(\mathbf{u}^r) - f(\mathbf{u}^r)) \geq \sigma/\theta^0 > 0,$$

which contradicts (15). As a result, we have (16) by noting (13).

Case 2: $\mathbf{u}^0 = \mathbf{y}^0$. For each r, let

$$\tau^r = \|\mathbf{u}^r - \mathbf{y}^r\|, \quad \mathbf{d}^r = \frac{1}{\tau^r}(\mathbf{u}^r - \mathbf{y}^r).$$

Then $\tau^r \to 0$, as $r \to \infty$, and we can assume that $\mathbf{d}^r \to \mathbf{d}^0$ for some unit vector $\mathbf{d}^0$. Using $\{\tau^r\}$ and $\{\mathbf{d}^r\}$, we see that

$$\lim_{r\to\infty} \frac{f(\mathbf{u}^r) - f(\mathbf{y}^r)}{\|\mathbf{u}^r - \mathbf{y}^r\|} = \lim_{r\to\infty} \frac{f(\mathbf{y}^r + \tau^r\mathbf{d}^r) - f(\mathbf{y}^r)}{\tau^r} \le \limsup_{\substack{\tau\searrow 0\\ \mathrm{y}\to\mathrm{u}^0\\ \mathrm{d}\to\mathrm{d}^0}} \frac{f(\mathbf{y} + \tau\mathbf{d}) - f(\mathbf{y})}{\tau}. \tag{18}$$

The last term of (18), called the semiderivative of f at $\mathbf{u}^0$ for $\mathbf{d}^0$, is known to be finite and equal to the one-sided directional derivative $f'(\mathbf{u}^0;\mathbf{d}^0)$ when f is convex and $\mathbf{u}^0 \in \mathrm{int}(\mathrm{dom} f)$ (see Chapters 7 – 9 of [15]). From these facts and Lemma 4.2, we have

$$\begin{aligned}
\lim_{r\to\infty} g^r(\boldsymbol{\omega}^r) &= \lim_{r\to\infty} \left(g^r(\mathbf{y}^r) + \|\boldsymbol{\omega}^r - \mathbf{y}^r\| \frac{g^r(\mathbf{u}^r) - g^r(\mathbf{y}^r)}{\|\mathbf{u}^r - \mathbf{y}^r\|} \right) \\
&= f(\mathbf{u}^0) + \|\boldsymbol{\omega}^0 - \mathbf{u}^0\| \lim_{r\to\infty} \frac{f(\mathbf{u}^r) - f(\mathbf{y}^r)}{\|\mathbf{u}^r - \mathbf{y}^r\|} \\
&\le f(\mathbf{u}^0) + \|\boldsymbol{\omega}^0 - \mathbf{u}^0\| f'(\mathbf{u}^0;\mathbf{d}^0) \le f(\mathbf{u}^0 + \|\boldsymbol{\omega}^0 - \mathbf{u}^0\|\mathbf{d}^0) = f(\boldsymbol{\omega}^0),
\end{aligned}$$

which, together with (13), implies (16). □

Corollary 4.4 *Assume that the sequence (12) is generated according to the extended ω-subdivision rule, and that $D \cap S^0 \neq \emptyset$. Then there exists an accumulation point $\boldsymbol{\omega}^0$ of the sequence $\{\boldsymbol{\omega}^i\}$ in D such that $f(\boldsymbol{\omega}^0) \ge f(\mathbf{x})$ for any $\mathbf{x} \in D \cap S^0$.*

Proof. Let $\{\mathbf{u}^{i_r}\}$ be a subsequence converging to some vertex of S^0, and abbreviate i_r to r. Then we can assume that $\boldsymbol{\omega}^r \to \boldsymbol{\omega}^0 \in D$, and besides

$$\lim_{r\to\infty} (g^r(\boldsymbol{\omega}^r) - f(\boldsymbol{\omega}^r)) = 0.$$

This implies that $f(\boldsymbol{\omega}^0) \ge f(\mathbf{x})$ for any $\mathbf{x} \in D \cap S^0$, because $g^r(\boldsymbol{\omega}^r) \ge f(\mathbf{x})$ for each r. □

5 Simplicial algorithm with the new subdivision rule

Let us summarize our algorithm incorporating the extended ω-subdivision rule. Given a tolerance $\varepsilon > 0$ and an integer $N > 0$, it can be described as a recursive algorithm:

```
algorithm simpl_bb($D, f, \varepsilon, N$)
    determine $n+1$ affinely independent points $\mathbf{v}_1, \ldots, \mathbf{v}_{n+1}$ such that $D \subset \mathrm{conv}\{\mathbf{v}_1, \ldots, \mathbf{v}_{n+1}\}$;
    let $\alpha \leftarrow -\infty$, and assign a null to $\mathbf{x}^\varepsilon$;
    call the procedure ext_omega$(1, \mathbf{v}_1, \ldots, \mathbf{v}_{n+1})$ and update $\mathbf{x}^\varepsilon$;
    return $\mathbf{x}^\varepsilon$;
end.

procedure ext_omega$(i, \mathbf{v}_1, \ldots, \mathbf{v}_{n+1})$
    $S^i \leftarrow \mathrm{conv}\{\mathbf{v}_1, \ldots, \mathbf{v}_{n+1}\}$;                              # bounding process
    compute the solution $[\mathbf{c}', c_0']$ to the system (6);
    solve the linear program $\mathrm{P}'(S^i)$ associated with $[\mathbf{c}', c_0']$;
    for an optimal solution $\boldsymbol{\omega}^i$ of $\mathrm{P}'(S^i)$, let $\beta(S^i) \leftarrow \mathbf{c}'\boldsymbol{\omega}^i + c_0'$;
    if $f(\boldsymbol{\omega}^i) > \alpha$ then
        $\alpha \leftarrow f(\boldsymbol{\omega}^i)$; $\mathbf{x}^\varepsilon \leftarrow \boldsymbol{\omega}^i$;
    end if
    if $\beta(S^i) - \alpha > \varepsilon$ then
        if $i \mod N \neq 0$ then                              # extended $\omega$-subdivision (i)
            compute the solution $\boldsymbol{\mu}'$ to the system (9), and let $J_+ \leftarrow \{j \mid \mu_j' > 0\}$;
            determine the subdivision point $\mathbf{u}$ of $S^i$ according to (10) and (11);
        else                                                   # extended $\omega$-subdivision (ii)
            choose a longest edge $\mathbf{v}_p$-$\mathbf{v}_q$ of $S^i$, let $\mathbf{u} \leftarrow (\mathbf{v}_p + \mathbf{v}_q)/2$ and $J_+ \leftarrow \{p, q\}$;
        end if
        for each $j \in J_+$ do                                # branching process
            call the procedure ext_omega$(i+1, \mathbf{v}_1, \ldots, \mathbf{v}_{j-1}, \mathbf{u}, \mathbf{v}_{j+1}, \ldots, \mathbf{v}_{n+1})$;
        end for
    end if
end.
```

Theorem 5.1 *If $\varepsilon > 0$, the algorithm* simpl_bb *terminates with a globally ε-optimal solution $\mathbf{x}^\varepsilon$ after finitely many calls of the procedure* ext_omega.

Proof. Suppose that simpl_bb does not terminate. Then it generates an infinite sequence of nested simplices, like (12). Let α^i denote the value of α at the end of the bounding process in the procedure ext_omega$(i, \mathbf{v}_1, \ldots, \mathbf{v}_{n+1})$, and let $\beta^i = \beta(S^i)$, which is identical to $g^i(\boldsymbol{\omega}^i)$ if g^i denotes the concave envelope of f on S^i. Also let $U = \max\{f(\mathbf{v}_j^1) \mid j = 1, \ldots, n+1\}$, where $\mathbf{v}_j^1$ is a vertex of S^1. Since $\{\alpha^i\}$ is nondecreasing and bounded from above by U, it converges to some $\alpha^0 \leq U$. Passing to a suitable subsequence, we have $\beta^r \to \beta^0$ and $\boldsymbol{\omega}^r \to \boldsymbol{\omega}^0$, as $r \to \infty$, where r is an abbreviation of i_r. From Theorem 4.3, we also see that $\beta^0 = f(\boldsymbol{\omega}^0)$ holds. However, $f(\boldsymbol{\omega}^i) \leq \alpha^i \leq \beta^i$ for every i, and hence $f(\boldsymbol{\omega}^r)$ and β^r both converge to α^0. Therefore, if $\varepsilon > 0$, the backtracking condition $\beta(S^i) - \alpha \leq \varepsilon$ holds for some i, say i_k, and consequently every nested sequence of simplices $\{S^i\}$ is finite. Since S^{i_k} is an n-dimensional simplex, the algorithm simpl_bb terminates after finitely many calls of ext_omega and partitions S^1 into $\{S^{i_k} \mid k = 1, \ldots, K\}$ for some finite number K. The backtracking condition guarantees the ε-optimality of $\mathbf{x}^\varepsilon$ on termination of the algorithm. $\square$

Numerical comparison

In the rest of this section, we report on numerical results of comparison between the algorithm simpl_bb and the usual simplicial algorithm according to an ordinary ω-subdivision rule. The test problem solved using those algorithms is a concave quadratic maximization problem of the form:

$$\left| \begin{array}{ll} \text{maximize} & f(\mathbf{x}) + \theta \mathbf{dy} \\ \text{subject to} & \mathbf{Ax} + \mathbf{By} \leq \mathbf{b}, \quad [\mathbf{x}, \mathbf{y}] \geq \mathbf{0}, \end{array} \right. \tag{19}$$

where

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\mathsf{T}\mathbf{Qx} + \mathbf{cx}.$$

To make the feasible set bounded, the vector $\mathbf{b} \in \mathbb{R}^m$ was fixed to $[1.0, \ldots, 1.0, n]^\mathsf{T}$ and all components of the last row of $\mathbf{A} \in \mathbb{R}^{m \times q}$ and $\mathbf{B} \in \mathbb{R}^{m \times (n-q)}$ were set to 1.0. Other entries of $\mathbf{A}$ and $\mathbf{B}$, together with those of $\mathbf{c} \in \mathbb{R}^q$ and $\mathbf{d} \in \mathbb{R}^{n-q}$, were generated randomly in the interval $[-0.5, 1.0]$, so that the percentages of zeros and negative numbers were about 20% and 10%, respectively. The matrix $\mathbf{Q} \in \mathbb{R}^{q \times q}$ was symmetric, positive semidefinite, tridiagonal, and the tridiagonal entries were random numbers in $[0.0, 1.0]$.

Note that the objective function of (19) can be linearized by replacing only the nonlinear part f with its concave envelope over a simplex. Therefore, we may implement the branching process in the $\mathbf{x}$-space of dimension $q \leq n$, instead of in the whole space of dimension n. Based on this decomposition principle [8], we coded the algorithm simpl_bb in GNU Octave 4.0.0 [4]. Other than this, the program code differs from the above description in two points: (i) it searches the children of S^i in decreasing order of their values of β for each i, and (ii) it uses the following as the backtracking condition, instead of $\beta(S^i) - \alpha > \varepsilon$,

$$\beta(S^i) - (1 + \varepsilon)\alpha > 0, \tag{20}$$

where ε was set to 10^{-5}. The alteration (i) is just a heuristic aiming to improve the lower bound α rapidly, and (ii) was made to prevent the magnitude of the optimal value from affecting the convergence of the program code. The number N prescribing the frequency of bisection in the extended ω-subdivision rule was fixed at 50. We also wrote the code for the revised simplex algorithm to solve $\mathrm{P}'(S^i)$, without using any optimization tools provided in Octave. To compare the performance with other algorithms, we coded the standard simplicial algorithm in two ways, one of which chooses the successor S^{i+1} to S^i in best-bound-first order, referred to as std_bound, and the other in depth-first order, referred to as std_depth. Both adopted the

Table 1: Comparison between simpl_bb and std_depth when $\theta = 5.0$.

$m \times n$		$q = 0.3n$		$q = 0.4n$		$q = 0.5n$		$q = 0.6n$	
		#	*time*	#	*time*	#	*time*	#	*time*
60×150	simpl_bb	4.2	0.0332	36.5	0.1768	73.1	0.2736	542.5	5.355
	std_depth	3.6	0.1064	17.7	0.4364	39.3	0.9712	968.7	22.79
90×150	simpl_bb	2.8	0.0628	29.5	0.2224	60.3	0.3048	350.7	4.179
	std_depth	2.8	0.1628	21.0	0.7812	40.4	1.5792	577.7	21.49
90×200	simpl_bb	6.3	0.0808	15.2	0.1512	59.6	0.6672	289.7	4.604
	std_depth	6.2	0.3384	13.2	0.7788	46.5	2.779	259.9	15.28
120×200	simpl_bb	2.5	0.1344	8.7	0.1616	17.4	0.2208	174.2	1.880
	std_depth	2.5	0.3700	8.8	0.8636	14.5	1.558	134.6	11.47
120×250	simpl_bb	1.9	0.1644	3.5	0.1876	78.3	1.322	341.4	11.94
	std_depth	1.9	0.4000	3.5	0.6740	25.3	3.188	523.5	61.88
150×250	simpl_bb	1.2	0.2640	33.0	0.6216	69.9	1.442	362.3	13.86
	std_depth	1.2	0.4940	31.6	4.877	49.1	8.103	342.8	58.06

regular ω-subdivision rule and the backtracking condition (20). As varying m, n, q and θ, we solved ten instances of (19) and measured the average performance of each code for each set of the parameters.

Figures 1 and 2 plot the changes in the average number of iterations and the average CPU time in seconds, respectively, taken by each program code when the dimension q of $\mathbf{x}$ was increased from 30 to 70 with (m, n, θ) fixed at $(60, 100, 5.0)$. Note that the number of iterations corresponds to the number of linear programs $\mathrm{P}'(S)$'s, or $\Pi(S)$'s, solved before termination of each program code. It behaves similarly for both simpl_bb and std_depth while std_bound requires much more iterations than those if q exceeds 40. When $q = 70$, the code std_bound took more than $1,000,000$ iterations for most instances, and so we gave up trying to measure its average performance. With respect to CPU time, simpl_bb is superior to std_depth considerably as well as to std_bound. This is considered because $\mathrm{P}'(S)$ is much easier to solve than $\Pi(S)$. Figures 3 and 4 show the results when the weight θ in the objective function was varied between 2.0 and 10.0 with $(m, n, q) = (60, 100, 30)$. When $\theta < 3.0$, the average performance of std_bound could not be obtained for the same reason as before. From these figures, we can again confirm the superiority of simpl_bb to the other two codes, especially in terms of CPU time.

The computational results for simpl_bb and std_depth on larger-scale instances are listed in Table 1, where the column labeled '#' gives the average number of iterations and the column

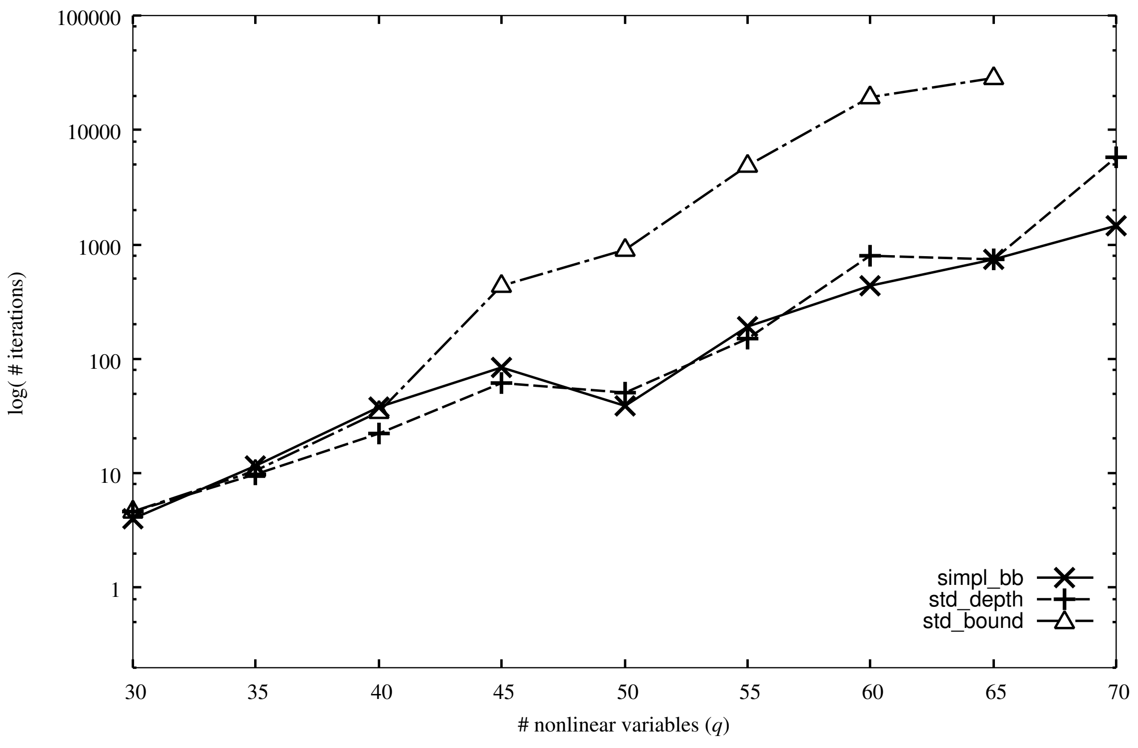

Figure 1: Number of iterations when $(m,n,\theta) = (60,100,5.0)$.

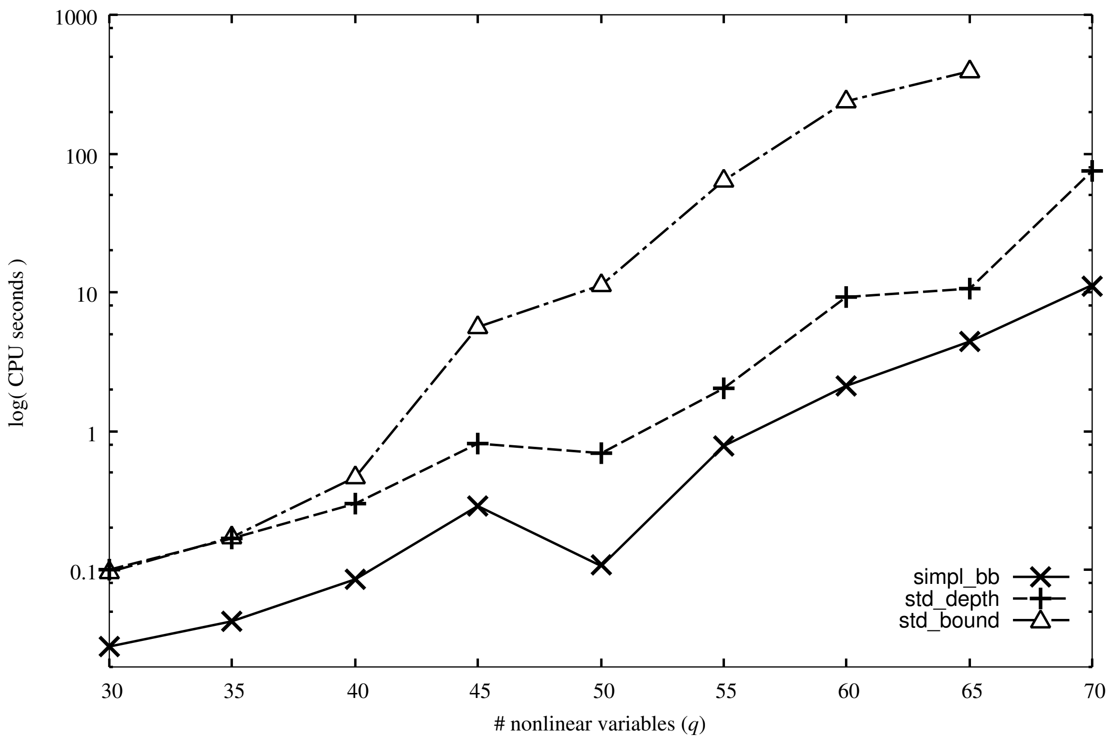

Figure 2: CPU time in seconds when $(m,n,\theta) = (60,100,5.0)$.

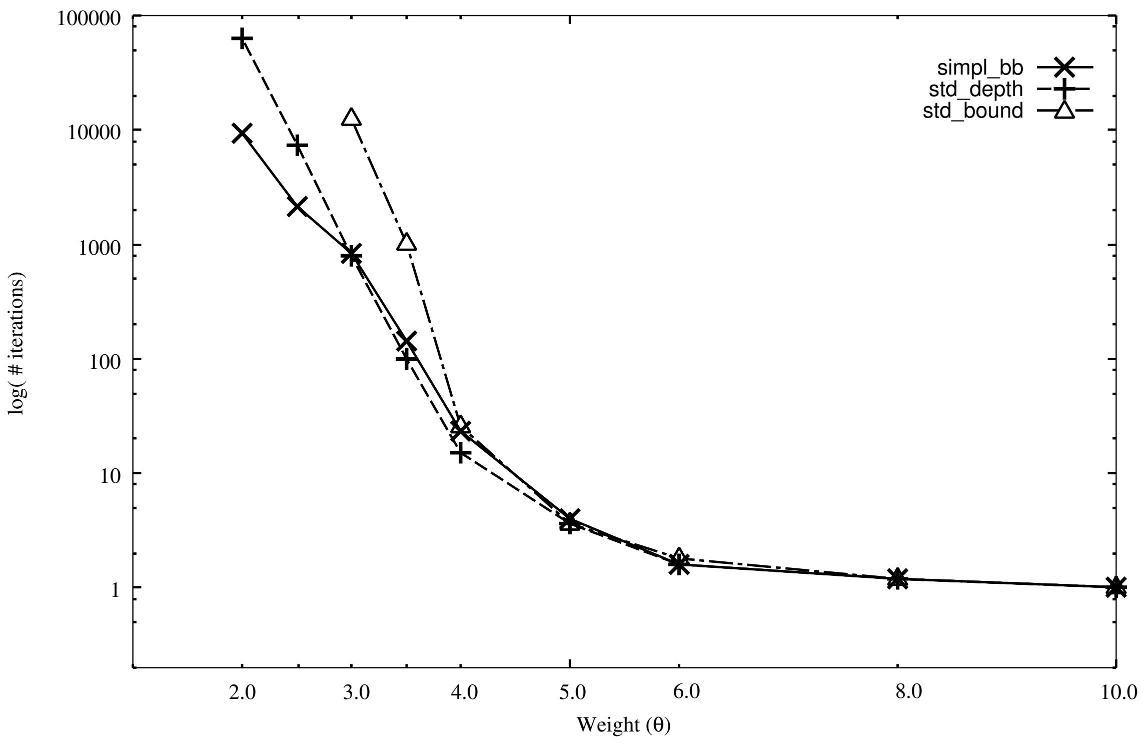

Figure 3: Number of iterations when $(m,n,q) = (60,100,30)$.

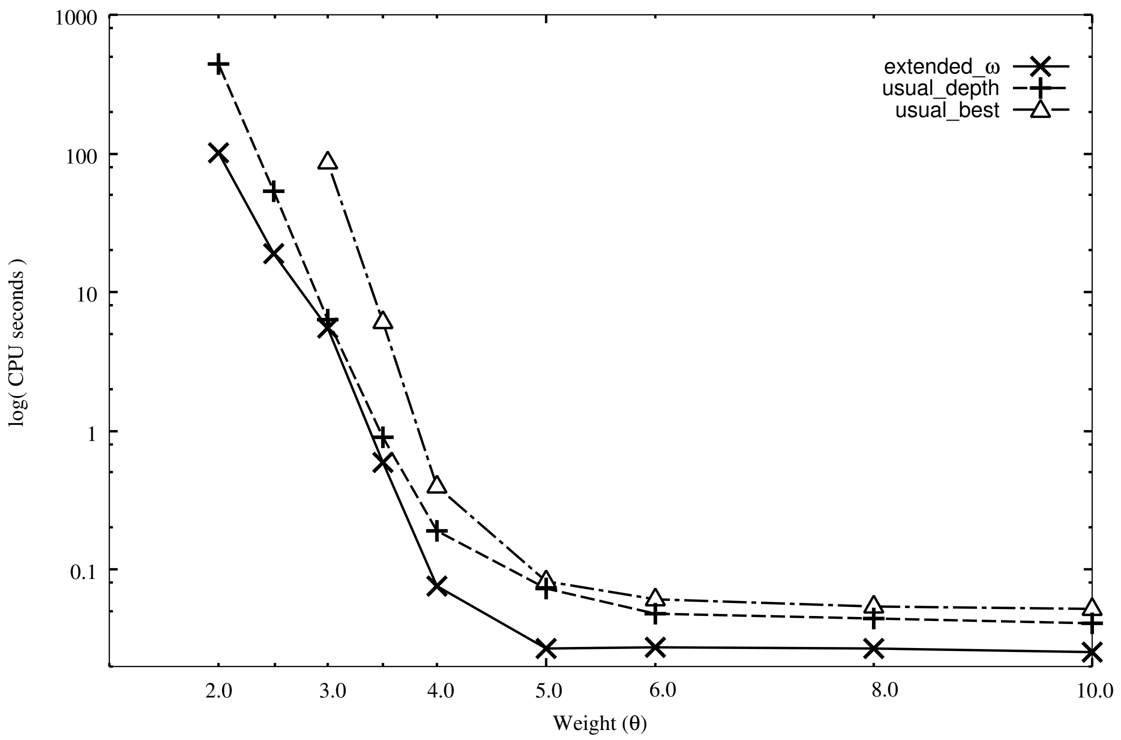

Figure 4: CPU time in seconds when $(m,n,q) = (60,100,30)$.

labeled '*time*' the average CPU time in seconds when (m,n,q) ranged up to $(150,250,150)$ with θ fixed at 5.0. Although simpl_bb might be a little inferior in the number of iterations, its average CPU time is far less than that of std_depth for every (m,n,q). This tendency is more noticeable when the proportion of nonlinear variables q/n is larger.

6 Concluding remarks

In the usual simplicial algorithm, the linear program $\Pi(S)$ is solved repeatedly for different simplices S. We have proposed to simplify $\Pi(S)$ into $\mathrm{P}'(S)$ whose feasible set does not depend on S. As pointed out in Section 3, there are three obvious disadvantages (a), (b) and (c) in adopting $\mathrm{P}'(S)$ instead of $\Pi(S)$. We have shown that the second point (b) can be offset by choosing the successor of S in depth-first order. To overcome the third point (c), we have developed a new rule for subdividing S, named extended ω-subdivision, and shown that the algorithm converges even under this subdivision rule. With regard to the first point (a), we have not taken any measurements. However, our numerical results provided in Section 5 indicate that we need not worry much about (a). One possible reason may be that the lower bound α given by the optimal solution $\boldsymbol{\omega}'$ of $\mathrm{P}'(S)$ is strong enough to compensate for the deterioration of the upper bound $\beta(S)$. Since $\boldsymbol{\omega}'$ is a vertex of the feasible set D, we could further strengthen the lower bound α by evaluating the value of f at vertices of D adjacent to $\boldsymbol{\omega}'$. In the succeeding paper, we will discuss such local search procedures which can be incorporated into the algorithm simpl_bb.

Acknowledgment

The author would like to thank the anonymous referees for their valuable comments, which significantly improved the readability of this article.

References

[1] Borovikov, V., "On the intersection of a sequence of simplices" (Russian), *Uspekhi Matematicheskikh Nauk* **7** (1952), 179–180.

[2] Chvátal, V., *Linear Programming*, W.H. Freeman (New York, 1983).

[3] Falk, J.E., and R.M. Soland, "An algorithm for separable nonconvex programming problems", *Management Science* **15** (1969), 550–569.

[4] GNU Octave, http://www.gnu.org/software/octave/.

[5] 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.

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

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

[8] Horst, R., and H. Tuy, *Global Optimization: Deterministic Approaches*, 3rd ed., Springer-Verlag (Berlin, 1996).

[9] Kuno, T., and P.E.K. Buckland, “A convergent simplicial algorithm with ω-subdivision and ω-bisection strategies”, *Journal of Global Optimization* **52** (2012), 371–390.

[10] Kuno, T., and T. Ishihama, “A generalization of ω-subdivision ensuring convergence of the simplicial algorithm”, *Computational Optimization and Applications* **64** (2016), 535–555.

[11] 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.

[12] Locatelli, M., and U. Raber, “Finiteness result for the simplicial branch-and-bound algorithm based on ω-subdivisions”, *Journal of Optimization Theory and Applications* **107** (2000), 81–88.

[13] Locatelli, M., and F. Schoen, *Global Optimization: Theory, Algorithms, and Applications*, SIAM (PA, 2013).

[14] Martin, R.K., *Large Scale Linear And Integer Optimization – A Unified Approach*, Kluwer (Dordrecht, 1999).

[15] Rockafellar, R.T., and R.J.-B. Wets, *Variational Analysis*, Springer-Verlag (Berlin, 1998).

[16] Sahni, S., “Computationally related problems”, *SIAM Journal on computing* **3** (1974), 262–279.

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

[18] Tuy, H., *Convex Analysis and Global Optimization*, Springer-Verlag (Berlin, 1998).