数理解析研究所講究録
第2027巻 2017年 167-178

On an extension of the ω-subdivision rule used in the simplicial algorithm for convex maximization

Takahito Kuno*

Graduate School of Systems and Information Engineering
University of Tsukuba, Tsukuba, Ibaraki 305-8573, Japan.

Abstract

In this paper, we discuss some difficulties of the simplicial algorithm detected in implementing under the ω-subdivision rule. To overcome those, we modify the bounding process and extend ω-subdivision. We also report numerical results for the simplicial algorithm according to this new subdivision rule.

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

1 Introduction

In 1976, Horst proposed a kind of branch-and-bound algorithm, named the simplicial algorithm, to solve convex maximization problems [3]. The algorithm now counted among the most popular algorithms in global optimization [4, 5, 14], along with the conical algorithm proposed by Tuy in 1964 [17]. In the branching process, whereas the latter uses cones, the simplicial algorithm uses a set of simplices to partition the feasible set; and in the bounding process, the algorithm estimates an upper bound of the objective function by maximizing its convex envelope on each simplex, which is further subdivided to refine the partition if the upper bound is large enough. As a rule for subdivision, Horst used a simple one which bisects each simplex across its longest edge, and gave a convergence proof for the algorithm. Although his proof contains a flaw, Thoai-Tuy [16] found and fixed it later, and also introduced the concept of exhaustiveness as a sufficient condition for the convergence. On the other hand, in the conical algorithm, Tuy utilized a byproduct of the bounding process to subdivide each cone without a guarantee of convergence. Even for the simplicial algorithm, we can adopt a similar rule, which subdivides each simplex radially around the maximum point of the concave envelope obtained as a byproduct in the bounding process. This so-called ω-subdivision rule is not exhaustive, and the convergence of both algorithms incorporating the rule remained open until Jaumard-Meyer [6] and Locatelli [11] proved it for the conical algorithm independently in 1998, 1999, and Locatelli-Raber [12, 13] did for the simplicial algorithm in 2000.

*The author was partially supported by a Grant-in-Aid for Scientific Research (C) 16K00028 from Japan Society for the Promotion of Sciences. E-mail: takahito@cs.tsukuba.ac.jp

Preceding their proofs by nearly a decade, Tuy showed in [18] that the conical algorithm with ω-subdivision converges if a certain nondegeneracy condition holds (see also [5, 19]). Kuno-Ishihama [9] showed that a more moderate condition always holds and guarantees the convergence for the conical algorithm. In a similar way, Kuno-Buckland [8] and Kuno-Ishihama [10] proved the convergence of the simplicial algorithm and generalized the ω-subdivision rule.

In this paper, we discuss some difficulties of the simplicial algorithm detected in implementing under the ω-subdivision rule. To overcome those, we modify the bounding process and extend the ω-subdivision rule. In Section 2, we define the target concave maximization problem and outline the usual simplicial algorithm for solving it. In Sections 3, we point out the difficulties of the algorithm and propose a technical solution. To cope with some emerging issues, in Section 4, we develop a new simplicial subdivision rule. Lastly, in Section 5, we report numerical results for the simplicial algorithm according to this rule.

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:

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

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

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

and assume that D is nonempty and bounded. We also assume that an n-simplex $S^1 \subset \mathbb{R}^n$ with vertices $\mathbf{v}_1^1, \ldots, \mathbf{v}_{n+1}^1$ is given 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, (1) has at least one globally optimal solution $\mathbf{x}^* \in D$. In general, however, there are multiple locally optimal solutions, many of which are not globally optimal. To solve (1) rigorously, we need to enumerate them using techniques such as branch-and-bound. One of standard branch-and-bound approaches is the simplicial algorithm proposed by Horst in 1976 [3], outlined below.

Outline of the simplicial algorithm

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

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

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

where J^i is an index set such that $j \in J^i$ if $\mathbf{v}_1, \ldots, \mathbf{v}^i_{j-1}, \mathbf{u}, \mathbf{v}^i_{j+1}, \ldots, \mathbf{v}^i_{n+1}$ are affinely independent. Out of active descendants of S^1, 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). We refer to $\mathbf{u}$ as the *central point for subdivision* of S^i.

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 β of f on $D \cap S^i$ with the value α of f at the best known feasible solution. The value of β 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, and hence its maximum point $\boldsymbol{\omega}^i$ over $D \cap S^i$ can be obtained by linear programming. If $\alpha \geq \beta$ for $\beta = g^i(\boldsymbol{\omega}^i)$, 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 which need to be further examined.

If the algorithm does not terminate in a finite amount of time, it generates a 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 $\mathbf{u}$ 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 $\mathbf{x}^*$ of (1) if the successor S^{i+1} to S^i is chosen in *best-first order*, i.e., S^{i+1} is a simplex with the largest β 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 [8, 9, 12, 13]. 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 more efficiently than bisection [8]. Whichever rule is adopted, in order to make the algorithm converge to $\mathbf{x}^*$, the successor S^{i+1} to S^i needs to be chosen in best-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 requires an upper bound β for the subproblem of (1):

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

Assume that $D \cap S \neq \emptyset$. If not, β may be simply set to $-\infty$. Replacing f with its concave envelope g on S, the subproblem $\mathrm{P}(S)$ is relaxed into a linearized problem:

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

Although $\mathrm{Q}(S)$ is certainly linear programming in $\mathbb{R}^n$, to obtain the standard form, we need 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:

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

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

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

Let $\boldsymbol{\lambda}_S$ be an optimal solution to $\Pi(S)$. Then $\boldsymbol{\omega}_S = \mathbf{V}\boldsymbol{\lambda}_S$ solves $\mathrm{Q}(S)$, and β is given as the optimal value $g(\boldsymbol{\omega}_S) = \mathbf{d}\boldsymbol{\lambda}_S$ of $\mathrm{Q}(S)$ and $\Pi(S)$.

In the usual implementation of the simplicial algorithm, when S^{i+1} is chosen as the successor to 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 $\Pi(S^i)$ and $\Pi(S^{i+1})$ differ only in a column of their respective coefficient vector and matrix, and so this reoptimization needs little simplex pivots. If not, however, $\Pi(S^i)$ and $\Pi(S^{i+1})$ are substantially different, and it requires a large number of simplex pivots. An easy way to reduce it 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 $\mathbf{x}^*$ of (1), the algorithm still generates a globally ε-optimal solution in finite time for any tolerance $\varepsilon > 0$. In the rest of this paper, we develop a more drastic revision for the simplicial algorithm.

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

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

The coefficient vector $[\bar{\mathbf{c}}, \bar{c}_0]$ of the objective function is given as a solution to the system of linear equations:

$$[\mathbf{c}, c_0] \begin{bmatrix} \mathbf{V} \\ \mathbf{e} \end{bmatrix} = \mathbf{d}. \tag{5}$$

Since $D = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \leq \mathbf{b}\}$ is assumed to be nonempty and bounded, $\overline{\mathrm{Q}}(S)$ always has an optimal solution $\bar{\boldsymbol{\omega}}_S \in D$, for which we have

$$g(\bar{\boldsymbol{\omega}}_S) = \bar{\mathbf{c}}\bar{\boldsymbol{\omega}}_S + \bar{c}_0 \geq g(\boldsymbol{\omega}_S) \geq f(\mathbf{x}), \quad \forall \mathbf{x} \in D \cap S. \tag{6}$$

Therefore, $g(\bar{\boldsymbol{\omega}}_S)$ can serve as the upper bound β for $\mathrm{P}(S)$. Furthermore, whichever S^{i+1}

is chosen as the successor to S^i, problem $\overline{\mathrm{Q}}(S^{i+1})$ differs from $\overline{\mathrm{Q}}(S^i)$ in only the objective function, and can be reoptimized with a very few primal simplex pivots. The substitution of $\overline{\mathrm{Q}}(S)$ for $\Pi(S)$, however, has three obvious disadvantages:

(i) the upper bound β deteriorates in quality, as is indicated by (6);

(ii) an additional effort is needed to solve the system (5); and

(iii) the solution $\overline{\boldsymbol{\omega}}_S$ cannot be used as the central point for subdivision of S.

Among these disadvantages, (ii) might be negligible if the successor S^{i+1} to S^i is chosen in depth-first order. Unless S^i is pruned, the successor S^{i+1} is a child of S^i, and hence the associated matrix $\mathbf{V}^{i+1}$ differs from $\mathbf{V}^i$ in only a column, say $\mathbf{v}_j^{i+1}$. We can update the inverse of the coefficient matrix of (5) in time $O(n^2)$, using the Sherman-Morrison-Woodbury formula (see for detail, e.g., [15]), into

$$\mathbf{W}^{i+1} = \left(\mathbf{I} - \frac{1}{\mathbf{e}_j\mathbf{y}}\left(\mathbf{y} - \mathbf{e}_j^{\mathsf{T}}\right)\mathbf{e}_j\right)\mathbf{W}^i, \tag{7}$$

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

$$\mathbf{W}^i = \begin{bmatrix} \mathbf{V}^i \\ \mathbf{e} \end{bmatrix}^{-1}, \quad \mathbf{y} = \mathbf{W}^i \begin{bmatrix} \mathbf{v}_j^{i+1} \\ 1 \end{bmatrix}.$$

Since the usual reoptimization procedure performs a single simplex pivot in time $O(m^2)$ using a formula similar to (7), the computational burden for solving (5) would be offset by solving $\overline{\mathrm{Q}}(S)$ instead of $\Pi(S)$, as long as n is not extremely larger than m.

4 Extension of the ω-subdivision rule

The easiest way to overcome disadvantage (iii) is to adopt the bisection rule for subdividing S^i. In this section, we develop a new subdivision rule which utilizes the optimal solution $\overline{\boldsymbol{\omega}}_S$ of $\overline{\mathrm{Q}}(S)$ more effectively.

Extended ω-subdivision

Once the system (5) has been solved, the following can be solved in $\boldsymbol{\mu}$ with a little additional effort:

$$\begin{bmatrix} \mathbf{V} \\ \mathbf{e} \end{bmatrix} \boldsymbol{\mu} = \begin{bmatrix} \overline{\boldsymbol{\omega}}_S \\ 1 \end{bmatrix}. \tag{8}$$

Let $\overline{\boldsymbol{\mu}}$ denote the solution to (8), and let $J_+ = \{j \mid \overline{\mu}_j > 0\}$. Then, by letting

$$\overline{\lambda}_j = \begin{cases} \overline{\mu}_j / \sum_{j \in J_+} \overline{\mu}_j & \text{if } j \in J_+ \\ 0 & \text{otherwise,} \end{cases} \tag{9}$$

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

$$\mathbf{u} = \mathbf{V}\overline{\boldsymbol{\lambda}}. \tag{10}$$

The rule for subdividing S radially around this point $\mathbf{u}$ is referred to as *extended ω-subdivision*. Since $\mathbf{e}\overline{\boldsymbol{\mu}} = 1$ holds, J_+ never vanishes, and hence $\mathbf{u}$ is well-defined through (8) – (10). However, if $\mathbf{u}$ falls in $\{\mathbf{v}_1, \ldots, \mathbf{v}_{n+1}\}$, 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 4.1 *Let $\mathbf{u}$ be defined through (8) – (10). If $\mathbf{u} \in \{\mathbf{v}_1, \ldots, \mathbf{v}_{n+1}\}$, then $f(\overline{\boldsymbol{\omega}}_S) \geq f(\mathbf{x})$ for any $\mathbf{x} \in D \cap S$.*

CONVERGENCE PROPERTIES

If there is a large difference in edge lengths of the simplex S, we cannot obtain stable solutions to the systems (5) and (8). To avoid such a case, we occasionally apply the usual bisection rule in the simplicial algorithm and shorten a longest edge of S by half. Let us suppose that this combination of extended ω-subdivision and bisection generates a sequence of nested simplices:

$$S^1 \supset S^2 \supset \cdots \supset S^i \supset S^{i+1} \supset \cdots, \tag{11}$$

where S^{i+1} is a child of S^i and shares n vertices with S^i. Let us denote $S^0 = \bigcap_{i=1}^{\infty} S^i$, which is an m-simplex with $m \leq n$ [1]. We have the following lemma, known as the *basic simplicial subdivision theorem* (see the textbook [19] for a proof):

Lemma 4.2 *For each i, let $\mathbf{x}^i$ be a point of S^i. For the sequence (11), assume that*

(i) for infinitely many i, simplex S^{i+1} is obtained from S^i through bisection, and

(ii) for all other i, simplex S^{i+1} is obtained through radial subdivision around $\mathbf{x}^i$.

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

Let $\mathbf{u}^i$ be the central point for subdivision of S^i defined by (8) – (10). Lemma 4.2 implies that the sequence $\{\mathbf{u}^i\}$ has an accumulation point in the vertex set of S^0 if we perform bisection infinitely many times to generate (11). In addition to this, we can prove the following, which guarantees the convergence of the algorithm under the extended ω-subdivision rule:

Theorem 4.3 *For the sequence (11), assume that*

(i) for infinitely many i, simplex S^{i+1} is obtained from S^i through bisection, and

(ii) for all other i, simplex S^{i+1} is obtained through radial subdivision around $\mathbf{u}^i$.

Then there exists an accumulation point $\overline{\boldsymbol{\omega}}^0$ of the sequence $\{\overline{\boldsymbol{\omega}}_{S^i}\}$ in D such that $f(\overline{\boldsymbol{\omega}}^0) \geq f(\mathbf{x})$ for any $\mathbf{x} \in D \cap S^0$.

ALGORITHM DESCRIPTION

Let us summarize the algorithm incorporating the extended ω-subdivision rule. For a given tolerance $\varepsilon > 0$ and a number $N > 0$, it is described as follows:

algorithm extended_omega(D, f, ε, N)
  determine the initial simplex $S^1 = \mathrm{conv}\{\mathbf{v}_j^1 \mid j = 1, \ldots, n+1\} \supset D$;
  $\mathscr{P} \leftarrow \emptyset$; $\mathscr{T} \leftarrow \{S^1\}$; $\mathbf{x}^1 \leftarrow \mathbf{0}$; $\alpha^1 \leftarrow -\infty$; $i \leftarrow 1$; $k \leftarrow 1$; *stop* $\leftarrow$ *false*;
  while *stop* = *false* do
    for each simplex $S \in \mathscr{T}$ do
      solve the system (5) and obtain its solution $[\overline{\mathbf{c}}, \overline{c}_0]$;
      solve the linear program $\overline{\mathrm{Q}}(S)$ associated with $[\overline{\mathbf{c}}, \overline{c}_0]$;
      for an optimal solution $\overline{\boldsymbol{\omega}}_S$ of $\overline{\mathrm{Q}}(S)$, let $\beta_S \leftarrow \overline{\mathbf{c}}\,\overline{\boldsymbol{\omega}}_S + \overline{c}_0$;
      if $f(\overline{\boldsymbol{\omega}}_S) > \alpha^i$ then
        $\mathbf{x}^i \leftarrow \overline{\boldsymbol{\omega}}_S$; $\alpha^i \leftarrow f(\mathbf{x}^i)$;
      end if
    end for
    sort simplices in $\mathscr{T}$ in increasing order of β_S and renumber them from k;
    $\mathscr{P} \leftarrow \{S \in \mathscr{P} \cup \mathscr{T} \mid \beta_S - \alpha^i \geq \varepsilon\}$; $\mathscr{T} \leftarrow \emptyset$;
    if $\mathscr{P} = \emptyset$ then
      stop $\leftarrow$ *true*;
    else
      select $S = \mathrm{conv}\{\mathbf{v}_j \mid j = 1, \ldots, n+1\}$ with the largest index k in $\mathscr{P}$;
      $\mathscr{P} \leftarrow \mathscr{P} \setminus \{S\}$; $k \leftarrow k - 1$;
      if $i \bmod N \neq 0$ then
        solve the system (8) to obtain its solution $\overline{\boldsymbol{\mu}}$, and let $J_+ \leftarrow \{j \mid \overline{\mu}_j > 0\}$;
        determine the central point $\mathbf{u}$ for subdivision of S according to (9) and (10);
      else
        choosing an longest edge $\mathbf{v}_i$–$\mathbf{v}_j$ of S, let $\mathbf{u} \leftarrow (\mathbf{v}_i + \mathbf{v}_j)/2$ and $J_+ \leftarrow \{i, j\}$;
      end if
      for each $j \in J_+$ do
        $S \leftarrow \mathrm{conv}\{\mathbf{v}_1, \ldots, \mathbf{v}_{j-1}, \mathbf{u}, \mathbf{v}_{j+1}, \ldots, \mathbf{v}_{n+1}\}$; $\mathscr{T} \leftarrow \mathscr{T} \cup \{S\}$;
      end for
    end if
    $\mathbf{x}^{i+1} \leftarrow \mathbf{x}^i$; $\alpha^{i+1} \leftarrow \alpha^i$; $i \leftarrow i + 1$;
  end while
  $\mathbf{x}^* \leftarrow \mathbf{x}^i$;
end.

From the observations so far, we can prove the following:

Theorem 4.4 *If $\varepsilon > 0$, the algorithm* extended_omega *terminates with an ε-optimal solution $\mathbf{x}^*$ of (1) after finitely many iterations.*

5 Numerical results

In this section, we report the numerical results comparing the extended ω-subdivision rule with other subdivision rules. The test problem solved using the simplicial algorithm incorporating those rules is a convex quadratic maximization problem of the form:

$$\left| \begin{array}{ll} \text{maximize} & f(\mathbf{x}) + \theta \mathbf{dy} \\ \text{subject to} & \mathbf{Ax} + \mathbf{By} \leq \mathbf{b}, \quad [\mathbf{x}, \mathbf{y}] \geq \mathbf{0}, \end{array} \right. \tag{12}$$

where

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

To make the feasible set bounded, the vector $\mathbf{b} \in \mathbb{R}^m$ was fixed to $[1.0, \ldots, 1.0, n]^{\top}$ and all components of the last rows 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 components 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, tridiagonal, and the tridiagonal entries were random numbers in $[0.0, 1.0]$.

Note that the objective function of (12) can be linearized by replacing only the nonlinear part f with its concave envelope. 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 [5], we coded the algorithm extended_omega in GNU Octave 4.0.0 [2], a numerical computing environment similar to MATLAB, tested it on one core of Intel Core i7 (4.00GHz). In order to compare the performance, we also coded the usual simplicial algorithm in two ways, one of which chooses the successor S^{i+1} of the current simplex S^i in best-first order, and the other does in depth-first order, as in the algorithm extended_omega. We refer to the former Octave code as usual_best, the latter as usual_depth, and the code of extended_omega as extended_ω. As the procedure for solving $\overline{\mathrm{Q}}(S)$, we used the revised simplex algorithm, which was not an optimization toolbox procedure but coded from scratch in Octave. Furthermore, we did not adopt the pruning criterion $\beta_S - \alpha^i \geq \varepsilon$ in each octave code, but instead used

$$\beta_S - (1 + \varepsilon)\alpha^i \geq 0,$$

where ε was set to 10^{-5}, so as to avoid the influence of the magnitude of the optimal value on the convergence. The number N prescribing the frequency of bisection in extended_omega was fixed at 50. As varying m, n, q and θ, we solved ten instances of (12) 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 Octave code when the dimensionality q of $\mathbf{x}$ increased from 30 to 70, with (m, n, θ) fixed at $(60, 100, 5.0)$. Figures 3 and 4 show the results when the weight θ in the objective function changed between 2.0 and 10.0, with $(m, n, q) = (60, 100, 30)$. We see from Figures 1 and 3 that extended_omega and usual_depth behave rather similarly and require less iterations than usual_best. However, Figures 2 and 4 indicate that extended_omega is rather faster than usual_depth in terms of CPU time. The computational results for extended_omega and usual_depth on larger-scale instances are sum-

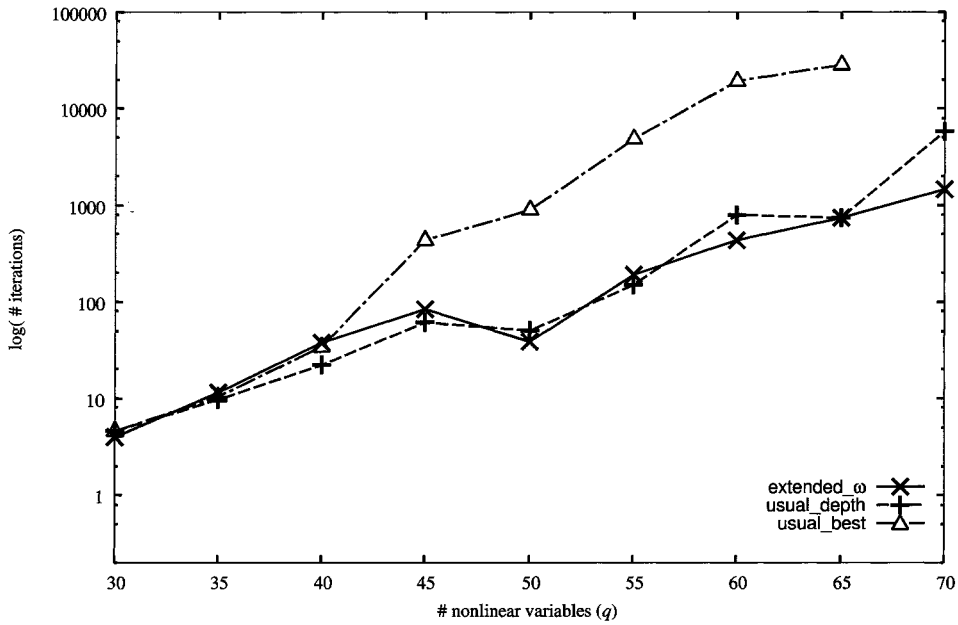

Figure 1: Number of iterations when $(m,n,\theta) = (60,100,5.0)$.

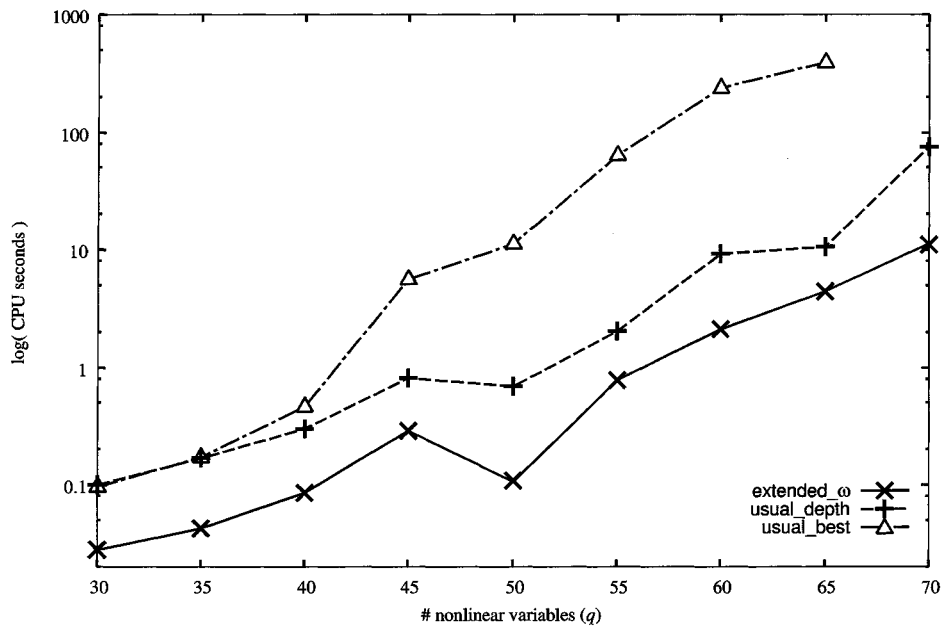

Figure 2: CPU time in seconds when $(m,n,\theta) = (60,100,5.0)$.

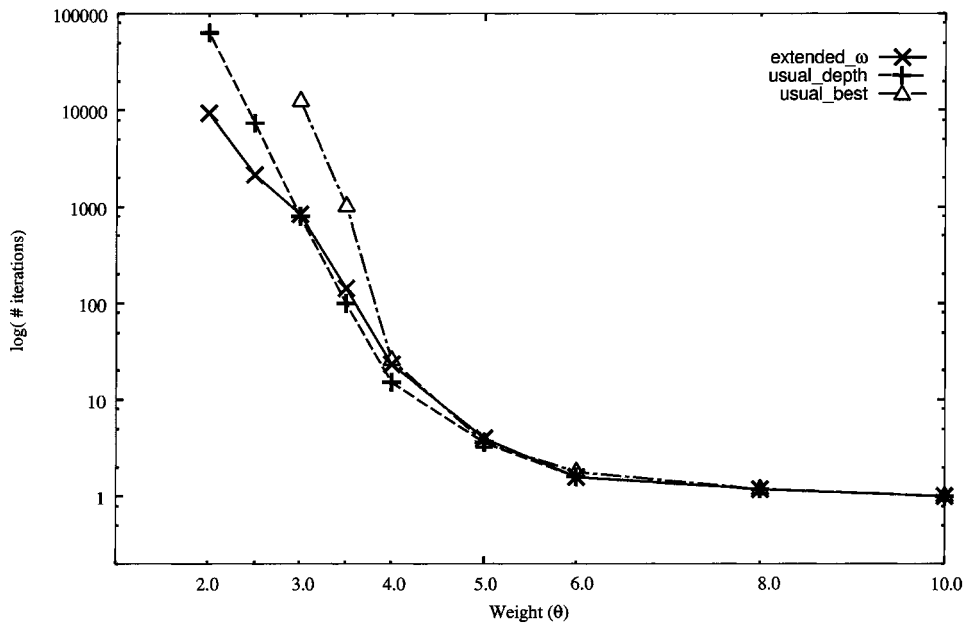

Figure 3: Number of iterations when $(m,n,q) = (60,100,30)$.

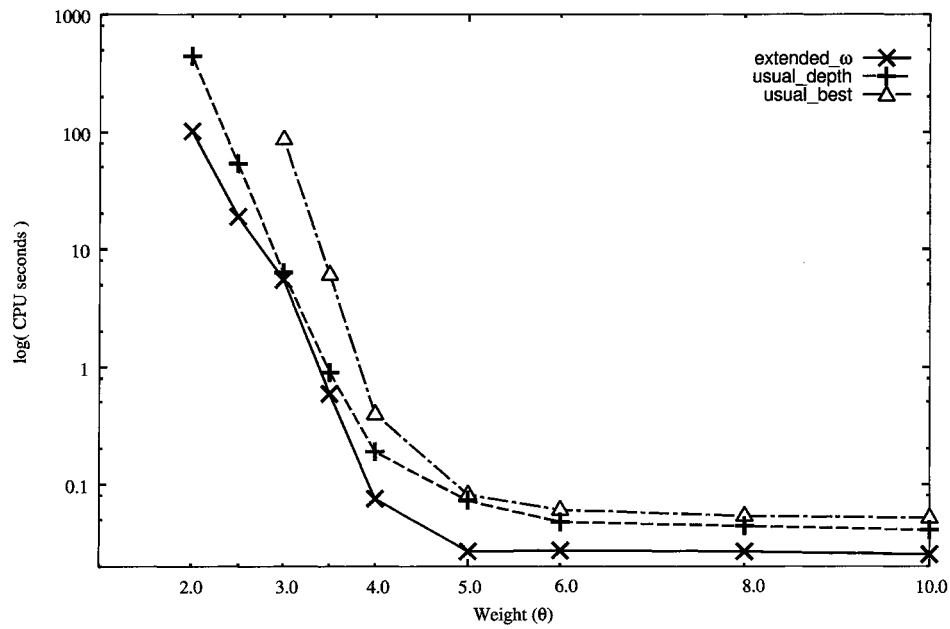

Figure 4: CPU time in seconds when $(m,n,q) = (60,100,30)$.

Table 1: Computational results of extended_omega 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	extended_ω	4.2	0.0332	36.5	0.1768	73.1	0.2736	542.5	5.355
	usual_depth	3.6	0.1064	17.7	0.4364	39.3	0.9712	968.7	22.79
90×150	extended_ω	2.8	0.0628	29.5	0.2224	60.3	0.3048	350.7	4.179
	usual_depth	2.8	0.1628	21.0	0.7812	40.4	1.5792	577.7	21.49
90×200	extended_ω	6.3	0.0808	15.2	0.1512	59.6	0.6672	289.7	4.604
	usual_depth	6.2	0.3384	13.2	0.7788	46.5	2.779	259.9	15.28
120×200	extended_ω	2.5	0.1344	8.7	0.1616	17.4	0.2208	174.2	1.880
	usual_depth	2.5	0.3700	8.8	0.8636	14.5	1.558	134.6	11.47
120×250	extended_ω	1.9	0.1644	3.5	0.1876	78.3	1.322	341.4	11.94
	usual_depth	1.9	0.4000	3.5	0.6740	25.3	3.188	523.5	61.88
150×250	extended_ω	1.2	0.2640	33.0	0.6216	69.9	1.442	362.3	13.86
	usual_depth	1.2	0.4940	31.6	4.877	49.1	8.103	342.8	58.06

marized in Table 1, where the column labeled '#' lists the average number of iterations and the column labeled '*time*' the average CPU time in seconds when (m,n,q) ranged up to $(150,250,150)$, with θ fixed at 5.0. For each particular (m,n,q), again, we see that extended_omega performs better than usual_depth, especially when the proportion of nonlinear variables q/n is relatively large.

These results suggest us not worry unnecessarily about the disadvantage (i) of $\overline{Q}(S)$ particularized in Section 3.

References

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

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

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

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

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

[6] Jaumard, B., and C. Meyer, "A simplified convergence proof for the cone partitioning algorithm", *Journal of Global Optimization* **13** (1998), 407–416.

[7] Jaumard, B., and C. Meyer, "On the convergence of cone splitting algorithms with ω-subdivisions", *Journal of Optimization Theory and Applications* **110** (2001), 119–144.

[8] 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.

[9] Kuno, T., and T. Ishihama, "A convergent conical algorithm with ω-bisection for concave minimization", *Journal of Global Optimization* **61** (2015), 203–220.

[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., "Finiteness of conical algorithm with ω-subdivisions", *Mathematical Programming A* **85** (1999), 593–616.

[12] 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.

[13] 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.

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

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

[16] Thoai, N.V., and H. Tuy, "Convergent algorithms for minimizing a concave function", *Mathematics of Operations Research* **5** (1980), 556–566.

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

[18] Tuy, H., "Normal conical algorithm for concave minimization over polytopes", *Mathematical Programming* **51** (1991), 229-245.

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