A METHOD FOR GLOBALLY MINIMIZING CONCAVE FUNCTIONS OVER CONVEX SETS

Karla Leigh HOFFMAN

The National Bureau of Standards, Washington, DC, U.S.A.

Received 1 February 1978
Revised manuscript received 17 April 1980

A method is described for globally minimizing concave functions over convex sets whose defining constraints may be nonlinear. The algorithm generates linear programs whose solutions minimize the convex envelope of the original function over successively tighter polytopes enclosing the feasible region. The algorithm does not involve cuts of the feasible region, requires only simplex pivot operations and univariate search computations to be performed, allows the objective function to be lower semicontinuous and nonseparable, and is guaranteed to converge to the global solution. Computational aspects of the algorithm are discussed.

Key words: Concave Programming, Extreme Point Solutions, Global Optimization, Non-convex Programming, Nonlinear Programming.

1. Introduction

The general problem which this paper addresses has the form

\[ \text{minimize } f(x), \]

\[ \text{over } S = \{ x: g_i(x) \leq 0, i = 1, \ldots, m \} \]

where \( f \) is a concave function defined throughout \( \mathbb{R}^n \) and \( g_i (i = 1, \ldots, m) \) are convex functions defined on \( \mathbb{R}^n \) whose gradients are continuous. It is assumed that \( S \) is compact and that a point in the strict interior of the feasible region exists.

Although many nonlinear programming algorithms will obtain a local solution to the problem, the principal difficulty with these methods is that the local minima which they obtain might not be global. A method for obtaining global solutions to separable nonconvex programs was first proposed by Falk and Soland [2], and extended by Soland [9]. McCormick [7] showed that the "convex envelope" approach could be applied to factorable functions. Horst [5] and Olsen [8] also develop techniques for solving concave minimizations over convex sets. However, all of the above-mentioned approaches require that a convex programming problem be solved at each step of an infinite iterative procedure.

This paper will present a method which incorporates many of the ideas of the
above methods while requiring only linear programming pivots and univariate searches to be performed at each iteration. The method does not require separability or factorability of the objective function or of the constraints, and is guaranteed to converge to a global solution.

The method is an extension of the Falk–Hoffman algorithm [1] for minimizing a concave function over a convex polyhedral set. Both algorithms enclose the feasible region in a polyhedron and generate a lower bound for the objective function by performing minimizations over that polyhedron. The lower bound is then refined by “tightening” the containing polyhedron. This process continues until the region enclosing the polyhedron is sufficiently close to the feasible region to exhibit a global solution to the original problem.

The principal disadvantages of the algorithm are that it also only has infinite convergence (although truncation is achieved by a tolerance criterion), and that as the number of iterations increases, the linear subproblems grow in size.

In Section 2, the details of the method are presented. Section 3 contains a small example to illustrate the method. Computational considerations are discussed in the final section.

2. The algorithm

Obtaining a global solution to concave minimization problems is more tractable than the minimization of general nonconvex functions because of the following well-known theorem.

**Theorem 1.** There exists an extreme point $x^*$ of the convex compact set $S$ which globally minimizes problem $P$.

However, for a nonlinear convex constraint set an infinite number of extreme points exist. The algorithm discussed in the paper will enclose the set within a polyhedron all of whose extreme points are known. Thus, a simple calculation of, and minimization over the extreme point functional values will determine a lower bound for problem $P$. If the minimizing point is feasible to problem $P$, then that point is also an upper bound and the problem is solved.

If not, the point obtained violates at least one constraint of problem $P$. A hyperplane of support to some violated constraint is constructed; it separates the set $S$ and that point. This hyperplane of support cuts through the original polytope, creating a new polytope which more tightly encloses the set $S$. The extreme points of this new set are obtained by pivoting operations, and a minimization of $f$ over the new set is easily performed.

Olsen [8] proposed adopting Kelley's cutting plane method for convex programming [6] to determine the hyperplanes of support with which to tighten the enclosing region. However, to obtain each hyperplane of support the solution of
a convex programming problem was necessary. In contrast, the method presented below will incorporate the ideas of Griffith and Stewart [3] and of Zoutendijk [11] to obtain a hyperplane of support by performing only one-dimensional searches.

The algorithm

Consider the problem:

\[ \begin{align*}
\text{minimize} & \quad f(x), \\
\text{P:} & \quad \text{subject to } g_i(x) \leq 0, \quad i = 1, \ldots, m
\end{align*} \]

where \( f \) is a concave function defined throughout \( \mathbb{R}^n \) and \( g_i \) (\( i = 1, \ldots, m \)) are convex functions whose gradients are continuous. Assume there is a point \( p \) in the strict interior of the feasible region, i.e. \( p \in \{ x : g_i(x) < 0 \quad i = 1, \ldots, m \} \). Let \( S = \{ x : g_i(x) \leq 0, \quad i = 1, \ldots, m \} \). We shall assume \( S \) is nonempty and compact.

Choose \( \varepsilon > 0 \) sufficiently small.

Step 0: Find a point \( p \) in the strict interior of \( S \), i.e.

\[ p \in \{ x : g_i(x) < 0 \quad i = 1, \ldots, m \}. \]

Set \( u^0 = f(p) \) (the best upper bound so far).

Find an enclosing linear polyhedron \( S^0 \) to the set \( S \) such that all vertices of \( S^0 \) are known, and which can be described as a set of constraints \( Ax \leq b \).

Let \( V^0 = \{ \text{the set of vertices of } S^0 \} \).

Set \( k = 1 \).

Go to step 1.

Step \( k \): (This step is entered with a polyhedron \( S^{k-1} \) enclosing \( S \), knowledge on the set \( V^{k-1} \), of extreme points of \( S^{k-1} \), and a current upper bound \( u^{k-1} \) for Problem P.)

Solve the Problem

\[ \begin{align*}
\text{P}^k: & \quad \text{minimize } f(x), \\
& \quad \text{subject to } x \in S^{k-1}
\end{align*} \]

by choosing \( \min\{f(v) : v \in V^{k-1}\} \).

Let \( x^k \) solve problem \( P^k \).

Now let \( \lambda_k \) solve the one-dimensional optimization problem

\[ \begin{align*}
\text{Q}^k: & \quad \text{minimize } \lambda, \\
& \quad \text{subject to } 0 \leq \lambda \leq 1, \quad x^k + \lambda(p - x^k) \in S.
\end{align*} \]

If \( \lambda_k = 0 \), then \( x^k \in S \) and is therefore the solution to problem \( P \); stop.
If $\lambda_k > 0$, set $z^k = x^k + \lambda_k (p - x^k)$. If $f(z^k) < u^{k-1}$, set $u^k = f(z^k)$; otherwise $u^k = u^{k-1}$. If $z^k - x^k < \varepsilon$, (or similarly, if $u^k - f(x^k) < \varepsilon$) stop; $x^k$ is the solution. If not, find all constraints of $S$ which are binding at $z^k$ and set

$$J^k = \{i : g_i(z^k) = 0; \ i = 1, \ldots, m\},$$

and choose any $j^k \in J^k$.

To the constraints of $S^{k-1}$, add the constraint

$$G^{j^k}_k(x) = g_{j^k}(z^k) + [\nabla g_{j^k}(z^k)]^T (x - z^k) \leq 0.$$

The equation-locus of the constraint $G^{j^k}_k$ is a hyperplane of support to $S$ at $z^k$, and this constraint cuts off the point $x^k$ from the set $S^k$ to be defined next. Set

$$S^k = S^{k-1} \cap \{x : G^{j^k}_k(x) \leq 0\}, \quad V^k = \{\text{extreme points of } S^k\}.$$

Set $k = k + 1$.

Return to step $k$.

The following lemmas are needed to prove convergence of the algorithm.

**Lemma 1.** The sequence of lower bounds $\{f(x^k)\}$ is monotonically nondecreasing.

**Proof.** The objective function $f$ is successively minimized over a nested decreasing sequence of sets $S^1, S^2, \ldots, S^k, S^{k+1}$. Thus

$$\min \{f(x) : x \in S^{k+1}\} \geq \min \{f(x) : x \in S^k\}$$

so that $f(x^{k+1}) \geq f(x^k)$.

**Lemma 2.** Let $\bar{x}$ be a limit point of the bounded sequence $\{x^k\}$. Let $\bar{x}$ be the solution of the one-dimensional optimization problem

$$\bar{Q}: \quad \text{minimize } \lambda,$$

subject to $\bar{x} + \lambda (p - \bar{x}) \in S, \quad 0 \leq \lambda \leq 1$.

Then $\bar{x}$ is a limit point of the sequence $\{\lambda^k\}$.

**Proof.** We have $\lim_{j \to \infty} x^{k(j)} = \bar{x}$ for some increasing sequence $\{k(j)\}$ of positive integers. The bounded sequence $\{\lambda_{k(j)}\}$ has a limit point $\lambda^*$; passing to a suitable sequence of $\{k(j)\}$, which for notational simplicity we denote again by $\{k(j)\}$, we have $\lim_{j \to \infty} \lambda_{k(j)} = \lambda^*$. The proof will be accomplished by showing that $\lambda^* = \bar{\lambda}$.

Since $\lambda_{k(j)}$ is a solution to problem $\bar{Q}^{k(j)}, \ x^{k(j)} + \lambda_{k(j)} (p - x^{k(j)})$ lies on the boundary of set $S = \{x : g_i(x) \leq 0, \ i = 1, 2, \ldots, m\}$. Thus, there exists at least one $r(j) \in \{1, 2, \ldots, m\}$ such that

$$g_{r(j)}[x^{k(j)} + \lambda_{k(j)} (p - x^{k(j)})] = 0.$$

Some member $r$ of the finite set $\{1, 2, \ldots, m\}$ must arise as $r(j)$ infinitely often;
passing to a further appropriate subsequence, still denoted \( \{k(j)\} \), we may assume

\[
g_i[x^{k(j)} + \lambda_{k(j)}(p - x^{k(j)})] = 0,
\]

which by continuity implies

\[
g_i[\bar{x} + \lambda^*(p - \bar{x})] = 0.
\]

Since \( 0 \leq \lambda_k \leq 1 \) implies \( 0 \leq \lambda^* \leq 1 \), and since the continuity of each \( g_i \) permits \( g_i[\bar{x} + \lambda^*(p - \bar{x})] \leq 0 \) to be inferred from \( g_i[x^{k(j)} + \lambda_{k(j)}(p - x^{k(j)})] \leq 0 \), it follows (using the convexity of \( S \) and \( p \in \text{interior of } S \)) that \( \lambda^* \) solves problem \( \bar{Q} \); hence \( \lambda^* = \bar{\lambda} \), as desired.

**Lemma 3.** Let \( z^k = x^k + \lambda_k(p - x^k) \). Then with \( \bar{x}, \bar{\lambda} \) as in Lemma 2, sequence \( \{z^k\} \) has \( \bar{z} = \bar{x} + \bar{\lambda}(p - \bar{x}) \) as a limit point.

**Proof.** By the proof of Lemma 2, there is a sequence \( \{k(j)\} \) such that \( x^{k(j)} \to \bar{x} \) and \( \lambda_{k(j)} \to \bar{\lambda} \) as \( j \to \infty \). It is a straightforward consequence that \( z^{k(j)} \to \bar{z} \) as \( j \to \infty \).

**Theorem 2.** Every limit point of the sequence \( x^k \) solves problem \( P \).

**Proof.** The algorithm yields a sequence of points \( \{x^k\} \) and a corresponding sequence of nondecreasing lower bounds \( \{f(x^k)\} \) (by Lemma 1). If at step \( k \), \( x^k \) is feasible to \( S \), the algorithm stops. If not, then as \( k \to \infty \), the sequence \( \{f(x^k)\} \) is monotonically non-decreasing and bounded from above by the value \( f(x^*) \) where \( x^* \) is any feasible solution to problem \( P \). But since the \( x^k \)'s are members of a compact set \( S^0 \), a point of accumulation \( \bar{x} \) must exist for the set \( \{x^k\} \). Let \( \{x^{k(j)}\} \) denote a subsequence of \( \{x^k\} \) which converges to \( \bar{x} \).

By the proof of Lemma 2, we may pass to a suitable subsequence such that \( \lambda_{k(j)} \to \bar{\lambda}, \; z^{k(j)} \to \bar{z}, \) and, for some \( r \in \{1, 2, \ldots, m\} \), \( g_r(z^{k(j)}) = 0 \). Since \( \lambda_{k(j)} > 0 \) (otherwise the algorithm would terminate), it follows that \( g_r(x^{k(j)}) > 0 \).

Assume \( \bar{x} \in S \) (which implies \( \bar{\lambda} > 0 \)). We claim that for some \( j \), the constraint \( G_r^{k(j)} \) imposed by the algorithm will cut off \( \bar{x} \), contradicting the assumption that \( x^{k(j)} \to \bar{x} \). To prove this claim, suppose to the contrary that

\[
G_r^{k(j)}(\bar{x}) = \nabla g_r(z^{k(j)})(\bar{x} - z^{k(j)}) \leq 0 \quad \text{for all } j.
\]

This implies, for

\[
G_r(x) = \nabla g_r(\bar{z})(x - \bar{z})
\]

that constraint \( G_r(z) \leq 0 \) does not cut off \( \bar{x} \) from \( S \). But since \( \bar{z} = \bar{x} + \bar{\lambda}(p - \bar{x}) \) where \( \bar{\lambda} \) solves problem \( \bar{Q} \) of Lemma 2, \( G_r \) must cut off \( \bar{x} \) for the same reason that \( G_{r}^j (j \in J^k) \) cuts off \( x^k \).

Thus \( \bar{x} \in S \). Since \( f(\bar{x}) \) is the limit of the nondecreasing sequence \( f(x^{k(j)}) \) of
lower bounds for problem P, \( \bar{x} \) solves problem P. It should be noted that since \( \bar{x} \in S, \bar{\lambda} = 0 \).

3. An example

The small example below was chosen to illustrate how the algorithm might perform on a nonseparable, highly nonlinear, problem. This algorithm performs best on problems where the solution occurs at the interaction of constraints since the solution of the linearized problem will be at a vertex. For that reason, an objective function was chosen such that the solution point would lie on the boundary of only one constraint.

Consider the problem:

\[
\begin{align*}
\text{minimize} \quad & f(x_1, x_2) = \frac{-(x_1^2 - 2x_1x_2 + x_2^2)}{2x_1} \\
\text{subject to} \quad & -28x_1 + 9x_2 + 21 \leq 0, \\
& 9x_1^2 - 72x_1 + 16x_2^2 \leq 0, \\
& x_1^2 + x_2^4 \leq 16, \\
& 64x_1^2 - 192x_1 - 36x_2 + 153 \leq 0.
\end{align*}
\]

Fig. 1 illustrates the feasible region \( S \) and the initial polyhedron \( S^0 \) chosen to enclose this region, namely

\[
\frac{1}{3} \leq x_1 \leq 3, \quad 0 \leq x_2 \leq 3.
\]

This region was chosen to illustrate what occurs when the enclosing polyhedron is not a tight fit to the feasible region. The convergence criterion \( \epsilon \) was chosen equal 0.00001.

A point interior to the feasible region is (2, 1.5). The vertex of the polyhedron \( S^0 \) having minimum objective value is \( (\frac{1}{3}, 3) \), with \( f(\frac{1}{3}, 3) = -6.25 \). The boundary point of \( S \) lying on the line segment connecting \( (\frac{1}{3}, 3) \) and \( (2, 1.5) \) is found to be \( z^1 = (1.4189, 2.0811) \).

Since \( f(1.41892, 2.08108) = -0.154503 < f(2, 1.5) = -0.0625 \), the new upper bound, \( u^1 \), is equal to \(-0.154503\).

Constraint (1) is binding at \( z^1 \) and since this constraint is linear,

\[
S^1 = S^0 \cap \{(x_1, x_2): -28x_1 + 9x_2 + 21 \leq 0, \}
\]

\[
J^1 = \{1\}.
\]

The new extreme points are \((1.71428, 3)\) and \((0.75, 0)\). Fig. 2 illustrates this new enclosing polytope.

Minimizing \( f \) over this polytope yields the solution point \((3, 0)\) which is infeasible. The point lying on the boundary between \((2, 1.5)\) and \((3, 0)\) is \( z^2 = (2.2196, 1.1706) \) with objective function value \(-0.24789\). Thus \( u^2 = -0.24789\).
Constraint (4) is binding at this point and the hyperplane of support is
\[ 92.1101x_1 - 36x_2 - 162.3072 \leq 0. \]

Thus
\[ S^2 = S^1 \cap \{(x_1, x_2): 92.1101x_1 - 36x_2 - 162.3072 \leq 0\}. \]

Fig. 3 illustrates this new enclosing polytope with its new extreme points 
\((2.9346, 3)\) and \((1.7621, 0)\). Their respective objective function values are 
\(-0.0007\) and \(-0.88105\).

Minimizing over this new polyhedron yields the minimizing point \((1.7621, 0)\).
Since this point is not feasible, the boundary point lying on the line segment
joining \((2, \frac{3}{2})\) and \((1.7621, 0)\) is computed, yielding the point \((1.8330, 0.4472)\).
\[ f(1.8330, 0.4472) = -0.5238917 = u^3. \]
and
\[ S^3 = S^2 \cap \{(x_1, x_2): 42.6266x_1 - 36x_2 - 62.0376 \leq 0\}. \]

Fig. 4 illustrates the enclosing polytope \(S^3\).

The algorithm continues for seven additional iterations before terminating at
the solution point \((1.66565, 0.2988)\) with objective function value \(-0.5608\).
The data relating to these iterations can be found in Table 1.
<table>
<thead>
<tr>
<th>Step $k$</th>
<th>Solution to $\min x^* \div f(x)$</th>
<th>New feasible point ${z^k}$</th>
<th>Supporting hyperplane</th>
<th>New extreme points</th>
<th>Best lower bound</th>
<th>Best upper bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$f(\frac{1}{3}, 3) = -6.25$</td>
<td>$(1.4189, 2.0811)$</td>
<td>$-28x_1 + 9x_2 + 21 \leq 0$</td>
<td>$(1.7143, 3)$</td>
<td>$-1.5$</td>
<td>$-0.0625$</td>
</tr>
<tr>
<td>2</td>
<td>$f(3, 0) = -1.5$</td>
<td>$(2.2196, 1.17059)$</td>
<td>$92.1101x_1 - 36x_2 - 162.3072 &lt; 0$</td>
<td>$(2.9346, 3)$</td>
<td>$-0.8811$</td>
<td>$-0.1545$</td>
</tr>
<tr>
<td>3</td>
<td>$f(1.7621, 0) = -0.8811$</td>
<td>$(1.8330, 0.4472)$</td>
<td>$42.6266x_1 - 36x_2 - 62.0376 &lt; 0$</td>
<td>$(1.4554, 0)$</td>
<td>$-0.7277$</td>
<td>$-0.2479$</td>
</tr>
<tr>
<td>4</td>
<td>$f(1.4554, 0) = -0.7277$</td>
<td>$(1.5476, 0.2540)$</td>
<td>$6.0941x_1 - 36x_2 - 0.2862 \leq 0$</td>
<td>$(1.6903, 0.2782)$</td>
<td>$-0.5899$</td>
<td>$-0.5239$</td>
</tr>
<tr>
<td>5</td>
<td>$f(1.6903, 0.2782) = -0.5899$</td>
<td>$(1.7015, 0.3222)$</td>
<td>$25.7869x_1 - 36x_2 - 32.2779 \leq 0$</td>
<td>$(1.7672, 0.3693)$</td>
<td>$-0.5672$</td>
<td>$-0.5406$</td>
</tr>
<tr>
<td>6</td>
<td>$f(1.6245, 0.2671) = -0.5676$</td>
<td>$(1.6253, 0.2793)$</td>
<td>$16.0403x_1 - 36x_2 - 16.0178445 \leq 0$</td>
<td>$(1.6649, 0.2959)$</td>
<td>$-0.5628$</td>
<td>$-0.5591$</td>
</tr>
<tr>
<td>7</td>
<td>$f(1.6649, 0.2959) = -0.5628$</td>
<td>$(1.6656, 0.2988)$</td>
<td>$21.2053x_1 - 36x_2 - 24.5598 \leq 0$</td>
<td>$(1.6469, 0.2878)$</td>
<td>$-0.5610$</td>
<td>$-0.5591$</td>
</tr>
<tr>
<td>8</td>
<td>$f(1.6666, 0.2640) = -0.5610$</td>
<td>$(1.609, 0.2711)$</td>
<td>$13.9337x_1 - 36x_2 - 12.6620 \leq 0$</td>
<td>$(1.6373, 0.2821)$</td>
<td>$-0.5609$</td>
<td>$-0.560802$</td>
</tr>
<tr>
<td>9</td>
<td>$f(1.6373, 0.2821) = -0.5609$</td>
<td>$(1.6378, 0.2837)$</td>
<td>$17.6330x_1 - 36x_2 - 18.6640 \leq 0$</td>
<td>$(1.6517, 0.2906)$</td>
<td>$-0.560846$</td>
<td>$-0.560802$</td>
</tr>
<tr>
<td>10</td>
<td>$f(1.6469, 0.2878) = -0.56086$</td>
<td>$(1.6472, 0.2885)$</td>
<td>$18.8370x_1 - 36x_2 - 20.6415 \leq 0$</td>
<td>$(1.6425, 0.2860)$</td>
<td>$-0.560842$</td>
<td>$-0.560802$</td>
</tr>
<tr>
<td>11</td>
<td>$f(1.6517, 0.2906) = -0.560842$</td>
<td>$(1.6518, 0.2910)$</td>
<td>$19.4328x_1 - 36x_2 - 21.6243 \leq 0$</td>
<td>$(1.6495, 0.2897)$</td>
<td>$-0.560825$</td>
<td>$-0.560825$</td>
</tr>
</tbody>
</table>
4. Computational considerations

In order to begin the algorithm, an initial enclosing polyhedron must either be computed or supplied. The simplest approach is to have the user supply upper and lower bounds on each of the variables. However, since efficiency increases if the polyhedron is a tight fit, other alternatives should be considered.

A tight enclosing polyhedron can be obtained by solving $2n$ convex programming problems of the form:

\[
\begin{align*}
\min & \quad x_i, \\
\text{s.t.} & \quad g_i(x) \leq 0; \quad i = 1, 2, \ldots, m; \\
\max & \quad x_i, \\
\text{s.t.} & \quad g_i(x) \leq 0; \quad i = 1, 2, \ldots, m
\end{align*}
\]

\((j = 1, 2, \ldots, n).\)

Alternatively, if the problem constraints include (or are known to imply) nonnegativity of the variables, then one convex programming problem of the form

\[
\max \sum_{i=1}^{n} x_i,
\]

\[
\text{s.t. } g_i(x) \leq 0; \quad i = 1, \ldots, m
\]

would produce an enclosing simplex described by \(\{x: \sum_j x_j \leq M, \ x_j \geq 0 \text{ for } j = 1, 2, \ldots, n\}\) where \(M\) is the optimal value of the convex program.

Although the first and second approaches will produce \(2^n\) and \(n + 1\) vertices, respectively, in general, not all of these vertices need to be stored. A feasible interior point must also be calculated before the algorithm can begin, and its function value supplies an upper bound on the problem; only vertices yielding an objective function value lower than this bound need be stored.

If these vertices are stored in increasing order of objective function value, then the minimization of the objective function is accomplished by merely choosing the vertex at the top of the list. This will yield a point \((x^k)\) which, if feasible, solves the problem. If not, a feasible point \((z^k)\) is then computed (problem \(P^k\)) using either Fibonacci search, the method of golden sections or some other efficient one-dimensional search technique.

The best upper bound is then computed, and any vertices having value greater than this bound are simply eliminated from the bottom of the list.

The most computationally expensive part of the algorithm involves generating the new vertices of the polyhedron created by adding an additional hyperplane. This is carried out in the following manner: Choose a \(j^k\) from \(J^k\).

(a) if \(v'\) is a vertex of \(S^{k-1}\) which satisfies the constraints \(G^k_j(x) \leq 0\), then \(v'\) is a vertex of \(S^k\), i.e., \(v' \in V^k\);

(b) if \(v'\) is a vertex of \(S^{k-1}\) which satisfies \(G^k_j(x) > 0\) for some \(j^k \in J^k\), then generate all neighbors of \(v'\) in the polyhedron \(S^k = \{x: G^k_j(x) \geq 0\} \cap S^k\); if such a
neighbor satisfies $G_f^k(x) = 0$, then it is a member of $V^k$, otherwise it is not.

Summarizing, the vertices of $S^{k+1}$ are either vertices of $S^k$ or are neighbors in $S^{k+1}$ of vertices $v'$ of $S^k$. The Falk–Hoffman [1] paper assumes nondegeneracy and proves that all vertices of $S^{k+1}$ not in $S^k$ can be obtained in this manner. This paper will prove the result without imposing a nondegeneracy assumption. A number of definitions must first be presented:

**Preliminary Definitions.** Let $P$ be a polytope in $R^n$. A set $F \subset P$ is a face of $P$ if either $F = \emptyset$ or $F = P$ or if there exists a supporting hyperplane $H$ of $P$ such that $F = H \cap P$. A face of $P$ which is neither $\emptyset$ nor $P$ is called a proper face of $P$. A face consisting of exactly one point is a vertex. Two vertices are neighbors if there exists a face $F$ of the polytope $P$ containing them. A facet of $P$ is a maximal proper face of $P$.

**Theorem 3** (Gruenbaum [4, p. 27] or Stoer and Witzgall [10, p. 71]). Every face of polytope $P$ is an intersection of facets of $P$ containing that face.

**Theorem 4.** Let $F$ be a proper face of the polytope $P$ and $v$ a vertex of $F$. Then there exists a vertex $u$ of $P$ such that $u \not\in F$ and $v$ is a neighbor of $u$.

**Proof (by induction).** The theorem is obviously true for polytopes of dimension 1 and 2.

Assume the theorem is true for all polytopes of dimension $n - 1$ for $n > 2$. Now consider a polytope $P$ of dimension $n$. Let $v$ be a vertex of a proper face $F$. If $F = \{v\}$, then since $F$ is a proper face another vertex of $P$ exists. Any neighboring vertex $u$ of $v$ is a vertex such that $u \not\in F$, and the theorem is proven for $F = \{v\}$.

Now consider $F \neq \{v\}$. Since every face (and therefore every vertex) of the polytope $P$ can be expressed as the intersection of all facets containing it, $\{v\} = \bigcap F_i$ where $F_i$ are the facets of $P$ containing $v$. If all facets $F_i$ containing $v$ are facets of $F$, then $\bigcap F_i = F \neq \{v\}$. Thus there must exist an $F_i$ such that $F \cap F_i \neq F$. Let $F^1 = F \cap F_i$.

$F^1$ must have dimension less than $n$, since the intersection of two distinct hyperplanes of dimension $n$, has dimension at most $n - 1$. If $F^1 = \{v\}$, then $F_i$ is a face of $P$ which joins $v$ to a vertex $u$ of $P$ such that $u$ is not on the face $F$ (since a neighbor of $v$ in $F_i$ must also be a neighbor of $v$ in $P$).

If $F^1 \neq \{v\}$, then $F^1$ has dimension less than $n$ and since $F^1$ is a polytope of dimension $n - 1$, the result holds by our assumption that the theorem is true for polytopes of dimension $n - 1$. Thus, the theorem is proven for $F \neq \{V\}$.

This theorem implies that if $\bar{v}$ is a vertex of $S^{k+1}$ which satisfies $G_f^k(x) = 0$ (face $F$ of polytope $\bar{S}^{k+1}$), then $v$ is a neighbor in $\bar{S}^{k+1}$ of some vertex $v' \in V^k$ for which $G_f^k(x) > 0$. 
Given such a vertex \( v' \), which satisfies \( G^j(x) > 0 \) for some \( j \in J^k \), we may represent the vertex in tabular form as

\[
s + t = \bar{b} \quad (s, t \geq 0)
\]

where \( s \) and \( t \) contain the basic and non-basic variables of the linear system corresponding to the system of constraints of \( S^k \). Adding the constraint \( G^j(x) \leq 0 \) results in an equation which when added to the system \( s + t = \bar{b} \) displays a non-feasible basic solution. Performing dual pivots on all current non-basic variables in the new row will yield all neighbors of \( v' \) which satisfy \( G^j(x) = 0 \).

The simplex tableaux required to perform these pivot operations will increase in size—one constraint is added per iteration—as the algorithm progresses. However, this is the only aspect of the algorithm which requires much computer time or storage; and other computations merely involve updating the list of vertices, and performing one-dimensional searches. Thus, even if these simplex calculations are computationally expensive it is expected that they will be cheaper than the alternative of solving convex programming problems.

Details of the implementation of the algorithm and computational experience will be reported in a subsequent paper.

Acknowledgment

I am indebted to James Falk, Garth McCormick and Christoph Witzgall for valuable discussions and suggestions regarding this research.

References