BRANCH AND BOUND METHODS FOR
REVERSED GEOMETRIC PROGRAMMING

by

W. GOCHET\textsuperscript{1} and Y. SMEERS\textsuperscript{2,3}

Onderzoeksrapport Nr. 7507.

2. Department of Engineering and CORE, Université Catholique de Louvain.
3. We sincerely thank Mr. OOMS for computational assistance on section VII of this paper.
I. INTRODUCTION

A signomial geometric program can be defined as the minimization (maximization) of a "generalized" polynomial, i.e. a polynomial in several variables, over a set defined by (in)equalities on generalized polynomials. This extensive class of mathematical programs however is nonconvex; moreover they cannot be transformed into convex programs as in for instance the case for the well-known class of posynomial or prototype geometric programs, (i.e. minimization problems with all the coefficients of the terms of the polynomials positive and the inequalities defining the feasible region are of the "less than" - type) [4]

A branch and bound method is proposed here to find the global optimum of a signomial program. It uses a result of Duffin and Peterson [2] which indicates that every signomial program can be transformed into a "reversed" program, i.e. a minimization program with all coefficients of the terms of the polynomials positive but some of the constraints are of the type "larger than" (see section II for algebraic definitions). These constraints of the "larger than" type (reversed constraints) are approximated from the outside by a one-term polynomial which allows us to approximate the reversed program by a prototype program (which is convex up to a transformation of the variables). A branch and bound scheme is then used to improve on this approximation with convergence to the global optimum being guaranteed.

The idea of branch and bound for reversed geometric programs is not new: in [5] and [8] the general method of Falk and Soland [6], [9] for the solution of separable non convex programs is adapted to the special case of geometric programming.
A similar approach to what is proposed here, is given in Passy [7]; the main difference lies in the approximation of the reversed constraints but both approaches can be combined as we will indicate further in the paper.

II. TRANSFORMATION OF A SIGNOMIAL PROGRAM.

A signomial geometric program as defined in the introduction can always be written as:

Program (S) : \[ \inf \left[ p_0(t) - r_0(t) \right] \]
subject to \[ p_j(t) - r_j(t) \leq \theta_j, \quad j = 1, 2, \ldots, S \]
where \[ t = (t_1, t_2, \ldots, t_m) > 0 \]

Program (R) : Find the infimum of a posynomial \( g_0(t) \) subject to the constraints \[ g_k(t) \leq 1, \quad k = r + 1, r + 2, \ldots, \ r + p. \]

Following [2], program (S) can be transformed into a reversed program (R) of the form:
where
\[ g_k(t) = \sum_{i \in [k]} u_i(t), \quad k = 0, 1, 2, \ldots, r + p. \]
and
\[ u_i(t) = c_{i1} t_1^{a_{i1}} t_2^{a_{i2}} \cdots t_m^{a_{im}}, \quad i \in [k], \quad k = 0, 1, \ldots, r + p \]
with \( [k] \equiv \{ m_k, m_k + 1, \ldots, n_k \} \), \( k = 0, 1, \ldots, r + p \)
and
\[ 1 = m_0 < n_0, \]

\[ n_i + 1 \equiv m_{i+1} - n_{i+1}, \quad i = 0, 1, \ldots, p + r - 1 \]

The coefficients \( c_i \) are positive while the exponents \( a_{ij} \) are arbitrary
real numbers.

It should be pointed out that new variables are introduced in
transforming program (S) into program (R). For ease of notation however we
kept the vector of variables as \( t = (t_1, t_2, \ldots, t_m) \). For the proof of conver-
gence it is essential that all variables are bounded away from 0 and \( +\infty \) in
an optimal sequence (canonical programs). Hence we make the following
assumption which actually can be slightly relaxed but a discussion of canoni-
cality of geometric programs would lead us too far from the main topic of
this paper (see [1]).

Assumption: There exists \( L_j > 0 \) and \( U_j > 0 \) such that

\[ L_j \leq t_j \leq U_j \]

It can be easily checked from the transformation that if such bounds exist
for the original variables in program (S), similar bounds also exist for the variables introduced by the transformation to program (R).

By properly introducing some more variables program (R) itself can be transformed \[3\] into a reversed geometric program with reversed constraints that have only two terms. Such a program will be called program (T).

III. APPROXIMATION OF THE REVERSED CONSTRAINTS

This section discusses the basic approximation of the reversed constraints

\[ g_k(t) \equiv \sum_{[k]} u_i(t) \geq 1, \quad k = 1, 2, \ldots, r \]

by a one-term polynomial constraint of the form

\[ \prod_{[k]} \left[ u_i(t) \right]^{\alpha_i} \geq b_k, \quad k = 1, 2, \ldots, r. \]

For brevity we denote \( u_i(t) \) simply \( u_i \) in the remainder of the paper. We need the following lemma for the one-term polynomial constraint to be defined.

**Lemma 1**: Let \( u^1, u^2, \ldots, u^n \) be \( n \) different points belonging to the open simplex \( \{ u \mid \sum_{r=1}^n u_r = 1, \quad u_r > 0, \quad r = 1, \ldots, n \} \) and let \( v^k = (v^k_1, v^k_2, \ldots, v^k_n) \)

be defined by

\[ v^k_i = \log u^k_i, \quad i = 1, \ldots, n \]

\[ k = 1, \ldots, n \]

Then the points \( v^1, v^2, \ldots, v^n \) are linearly independent.

**Proof**: Consider \( \text{conv} \ \{ v^1, v^2, \ldots, v^n \} \) where \( \text{conv} \) denotes convex hull. By
contradiction assume that \( v^1, v^2, \ldots, v^n \) are linearly dependent.

Then, \( v^k = \sum_{i \in I} \lambda_i v^i \) for some \( k \in \{1, 2, \ldots, n\} \) and \( I \subseteq \{1, 2, \ldots, n\} \) \( \setminus \{k\} \)

with \( \sum_{i \in I} \lambda_i = 1 \) and \( \lambda_i \geq 0, i \in I \)

Furthermore, at least two \( \lambda_i > 0, i \in I \), since \( v^i \neq v^j, i \neq j, i \in I, j \in I \).

By the strict convexity of \( \sum_{i \in I} e^{v_i} \) it follows

\[
\sum_{j=1}^{n} u^k = \sum_{j=1}^{n} \frac{v^k}{v^j} = \sum_{j=1}^{n} \lambda_i \frac{v^i}{v^j} < \sum_{j=1}^{n} \sum_{i \in I} \lambda_i \frac{v^i}{v^j} = 1
\]

which is a contradiction.

O.E.D.

The condition stated in lemma 1 in fact guarantees the existence of a surface

\[
\prod_{i=1}^{n} v_i^{\alpha_i} = \beta \text{ containing } u^1, u^2, \ldots, u^n.
\]

Indeed such a surface exists if there exists \( \alpha^1, \alpha^2, \ldots, \alpha^n, \beta \) satisfying.

\[
(1) \quad \sum_{i=1}^{n} \alpha_i \log u_i^k = \log \beta, \quad k = 1, \ldots, n
\]

The coefficients \( \alpha_1, \alpha_2, \ldots, \alpha_n, \log \beta \) are only defined up to a constant factor. Henceforth, we assume \( \beta \) arbitrary in \((0,1)\). The \( \alpha_i \) are then uniquely defined under the conditions of lemma 1. The next lemma proves that, under certain conditions, the region \( \prod_{i=1}^{n} u_i^{k_i} \geq \beta \) is an approximation from the outside to a part of the region defined by the reversed constraint \( \sum_{i=1}^{n} u_i < 1 \).

**Lemma 2**: Let \( u^1, u^2, \ldots, u^n \) satisfy the conditions of lemma 1. Suppose that all \( \alpha_i \) obtained from (1) are nonnegative. Then
where \( C \equiv \{ u \in \mathbb{R}^n \mid u \geq \sum_{k=1}^{n} \lambda_k u_k^k; \sum_{k=1}^{n} \lambda_k = 1 \text{ and } \lambda_k \geq 0, \text{ all } k \} \)

and \( C_{\text{app}} \equiv \{ u \in \mathbb{R}^n \mid \prod_{i=1}^{n} u_i^{\alpha_i} \geq \beta \} \)

Proof: Let \( u \in C \). Then,

\[
\prod_{i=1}^{n} u_i^{\alpha_i} \geq \prod_{i=1}^{n} \left( \sum_{k=1}^{n} \lambda_k u_i^k \right)^{\alpha_i}
\]

or

\[
\sum_{i=1}^{n} \alpha_i \log u_i \geq \sum_{i=1}^{n} \alpha_i \log \left( \sum_{k=1}^{n} \lambda_k u_i^k \right) - \sum_{i=1}^{n} \alpha_i \sum_{k=1}^{n} \lambda_k \log u_i^k
\]

\[
= \sum_{k=1}^{n} \lambda_k \sum_{i=1}^{n} \alpha_i \log u_i^k = \sum_{k=1}^{n} \lambda_k \beta = \beta
\]

In the case of a two-term reversed constraint, lemma 2 is illustrated in figure 1.
The condition $\alpha_i > 0$, $\forall i$, also guarantees some regularity in the approximation of $C$ by $C_{\text{app}}$ in the sense to be explained below. Therefore we define the set $C_{\text{app}}$ as

$$C_{\text{app}} = \{ u \in \mathbb{R}^n | \sum_{i=1}^{n} u_i \leq 1 \text{ and } \prod_{i=1}^{n} u_i = \beta \}$$

To construct a useful measure of distance between the sets $C_{\text{app}}$ and $\{ u | \sum_{i=1}^{n} u_i = 1 \}$, consider the ray $\{ u \mid u = \lambda \vec{v}, \vec{v} \in C_{\text{app}} \}$. This ray intersects the hyperplane $\sum_{i=1}^{n} u_i = 1$ at the point $\lambda \vec{v}$ for which

$$\sum_{i=1}^{n} \lambda u_i = 1.$$

The distance between $C_{\text{app}}$ and $\{ u | \sum_{i=1}^{n} u_i = 1 \}$ is defined here as

$$\lambda_{\text{sup}} = \sup \{ \lambda | \sum_{i=1}^{n} \lambda u_i = 1, \vec{v} \in C_{\text{app}} \}$$

Ultimately we want to construct approximations (by appropriately choosing a sequence $u_1^{k}, u_2^{k}, \ldots, u_n^{k}$ in the simplex $\sum_{i=1}^{n} u_i = 1$, $u_i > 0$) such that $\lambda_k^{\text{sup}} \to 0$ as $k \to \infty$

To see that this condition is not satisfied if the points $u_1, u_2, \ldots, u_n$ are chosen such that $\alpha_i < 0$, for some $i \in \{1, 2, \ldots, n\}$, in the approximation $\prod_{i=1}^{n} u_i = \beta$, consider the points

$$u_1 = \left[ \frac{1}{2} - 2n^2, \left( \frac{1}{2} - n \right)^2, \left( \frac{1}{2} + n \right)^2 \right]$$

$$u_2 = \left[ \frac{1}{2}, \left( \frac{1}{2} \right)^2, \left( \frac{1}{2} \right)^2 \right]$$
\[ u^3 = \left[ \frac{1}{2} - 2\eta^2, \left( \frac{1}{2} + \eta \right)^2, \left( \frac{1}{2} - \eta \right)^2 \right] \]

Obviously all three points are on the simplex \( \sum_{i=1}^{3} u_i = 1, u_i \geq 0 \) for \( \eta > 0 \) and sufficiently small. The approximation can be checked to be (take i.e. \( \beta = \frac{1}{2} \)):

\[
\begin{align*}
& u_1^{-1/2} u_2^{1/2} u_3^{1/2} = \frac{1}{2} \\
& u_i > 0, \forall i
\end{align*}
\]

Note that the approximation is independent of \( \eta \).
Consider the point \((\varepsilon, \frac{1}{2} \varepsilon, \frac{1}{2} \varepsilon)\) satisfying \( u_1^{-1} u_2^{1/2} u_3^{1/2} = \frac{1}{2} \).

As \( \eta \to 0 \) this point always remains on the approximating surface and \( \lambda_{sup} \to 0 \); in fact \( \lambda_{sup} \to +\infty \) as \( \varepsilon \to 0 \).

The next lemma gives a condition under which \( \lambda_{sup} \) is finite.

**Lemma 3**: Let \( \{u^1, u^2, \ldots, u^n\} \) satisfy the conditions of lemma 1. Then \( \lambda_{sup} \) is finite iff \( u \) is chosen in such a way that \( \alpha_i > 0, i = 1, \ldots, n \) in the approximation \( \prod_{i=1}^{n} u_i = \beta \). Moreover \( \lambda_{sup} \) is attained at a finite point if \( \alpha_i > 0, i = 1, \ldots, n \).

**Proof**: \( \lambda_{sup} \) is given by the value of the following optimization program GP1 and its equivalent GP 2.

**Program GP1**: \( \sup \lambda \)

subject to:

\[
\begin{align*}
\lambda \sum_{i=1}^{n} u_i &= 1 \\
\prod_{i=1}^{n} u_i &= \beta \\
\sum_{i=1}^{n} \alpha_i u_i &= \beta \\
\sum_{i=1}^{n} u_i &= \frac{1}{\beta} \\
u_i &> 0, \forall i
\end{align*}
\]

**Program GP2**: \( \inf \sum_{i=1}^{n} u_i \)

s.t.

\[
\begin{align*}
\beta^{-1} \prod_{i=1}^{n} u_i &= \leq 1 \\
\sum_{i=1}^{n} \alpha_i u_i &= \leq 1 \\
u_i &> 0, \forall i
\end{align*}
\]
Program GP2 is a consistent prototype geometric program which must have positive infimum if $\lambda_{\text{sup}}$ has to be bounded. The constraints of the geometric dual of GP2 are:

$$\sum_{i=1}^{n} \delta_i = 1$$

$$\delta_i - \alpha_i (\delta_{n+1} - \delta_{n+2}) = 0, \; i = 1, \ldots, n$$

$$\delta_i \geq 0, \; i = 1, 2, \ldots, n + 2$$

Since GP2 is consistent its infimum is positive iff the dual is consistent (Th 1 p.166 in [4]). From the dual constraint set it is then clear that either

$$\alpha_i > 0, \; i = 1, \ldots, n$$

or

$$\alpha_i < 0, \; i = 1, \ldots, n$$

The second possibility can be ruled out by considering any of the points $u_j; j \in \{1, \ldots, n\}$. This point satisfies

$$\prod_{i=1}^{n} (u_j)^{\alpha_i} = \beta \text{ with } 0 < \beta < 1 \text{ and } 0 < u_j < 1, \; \forall i$$

which is impossible if $\alpha_i \leq 0, \; \forall i$.

The optimum in GP2 will be attained if the program is canonical (Th 1(ii), p. 173 in [4]). GP2 is canonical iff there exists a dual feasible solution with $\delta_i > 0, \; i = 1, \ldots, n + 2$.

From the dual constraint set this clearly means that

$$\alpha_i > 0, \; i = 1, \ldots, n$$

We will illustrate that the condition of strict positivity on the $\alpha_i$ does not guarantee that $\lambda^k_{\text{sup}} \rightarrow 0$ as $\lim_{k \rightarrow \infty} u_i^k = \bar{u}, \; \forall i$. 
Lemma 4: The set $\{u > 0 \mid \prod_{i=1}^{n} u_i^{\alpha_i} \geq \beta \text{, with } \alpha_i > 0\}$ is a strictly convex set.

Proof: We first prove that $U$ is convex by showing that $\prod_{i=1}^{n} u_i^{\alpha_i}$ is quasi-concave when $\alpha_i > 0$.

Take $\bar{u}$ and $\bar{u}$ such that

\[ \prod_{i=1}^{n} u_i^{\alpha_i} \geq \prod_{i=1}^{n} \bar{u}_i^{\alpha_i} \]

or

\[ \sum_{i=1}^{n} \alpha_i \log \bar{u}_i \geq \sum_{i=1}^{n} \alpha_i \log \bar{u}_i \]

Consider the point $\lambda \bar{u} + (1 - \lambda)\bar{u}$ with $0 \leq \lambda \leq 1$.

\[ \sum_{i=1}^{n} \alpha_i \log [\lambda \bar{u}_i + (1 - \lambda) \bar{u}_i] \geq \sum_{i=1}^{n} \alpha_i \log \bar{u}_i + (1 - \lambda) \log \bar{u}_i \]

or

\[ \prod_{i=1}^{n} [\lambda \bar{u}_i + (1 - \lambda) \bar{u}_i]^{\alpha_i} \geq \prod_{i=1}^{n} \bar{u}_i^{\alpha_i} \]

Hence, $\prod_{i=1}^{n} u_i^{\alpha_i}$ is quasi-concave for $\alpha_i > 0$.

To prove strict-convexity of the set, take $\bar{u}$ and $\bar{u}$ such that

\[ \prod_{i=1}^{n} u_i^{\alpha_i} = \beta \]

and

\[ \prod_{i=1}^{n} \bar{u}_i^{\alpha_i} = \beta \]
Since \( \sum_{i=1}^{n} \alpha_i \log \left[ \lambda \bar{u}_i + (1 - \lambda) \bar{u}_i \right] > \lambda \sum_{i=1}^{n} \alpha_i \log \bar{u}_i \)

\[ + (1 - \lambda) \sum_{i=1}^{n} \alpha_i \log \bar{u}_i \text{ for } 0 < \lambda < 1 \]

it follows \( \sum_{i=1}^{n} \alpha_i \log \left[ \lambda \bar{u}_i + (1 - \lambda) \bar{u}_i \right] > \log \beta \)

which implies \( \prod_{i=1}^{n} \left[ \lambda \bar{u}_i + (1 - \lambda) \bar{u}_i \right]^{\alpha_i} > \beta \)

Q.E.D.

We given an example (see figure 2) illustrating that \( \lambda_k \) may be bounded away from 0 when \( \lim_{k \to \infty} u_{i,k} = \bar{u}, \forall i, \) and \( u_{i,k} \) chosen in such a way that \( \alpha_i > 0 \)

in the approximation \( \prod_{i=1}^{n} u_{i}^{\alpha_i} > \beta . \)

\[ (0,0,1) \]

\[ (0,1,0) \]

\[ (1,0,0) \]

Figure 2.
In figure 2 the intersection of \( \prod_{i=1}^{n} u_i = \beta \) with the simplex is given by the dotted curves (see lemma 4) and \( \lambda \sup \) is some finite positive number, say \( \lambda^x \). Consider the points \( u^{-1}, u^{-2}, \) and \( u^{-3} \).

They clearly satisfy the conditions of lemma 1 and \( \prod_{i=1}^{n} u_i = \beta \) obviously has the same intersection with the simplex as \( \prod_{i=1}^{n} u_i = \beta \). Hence \( \lambda \sup = \lambda^x \).

Letting \( u^{-1} \) and \( u^{-3} \) go to \( u^{-2} \) along the intersection curve shows that \( \lambda \sup \) remains a positive constant even though the points on the simplex collapse.

We now demonstrate how a sequence \((u^1, k, u^2, k, \ldots, u^n, k)\) can be constructed such that

\[
\lim_{k \to \infty} \lambda^k \sup = 0 \text{ as } (u^1, k, u^2, k, \ldots, u^n, k) \to (\bar{u}, \bar{u}, \ldots, \bar{u})
\]

where the points \((u^1, k, u^2, k, \ldots, u^n, k)\) satisfy the conditions of lemma 1 for each \( k \).

For that, every set of points in the sequence is taken as the set of extreme points of the polyhedron \( Q(a) \) defined by:

\[
Q(a) \equiv \{ u \mid u_i \geq a_i, \forall i \text{ and } \sum u_i = 1 \} \text{ where } a_i > 0, \forall i
\]

and \( \sum_{i=1}^{n} a_i < 1 \)

An example of such a polyhedron is given in figure 3.

The extreme points of the polyhedron \( Q(a) \) are given by
Figure 3.

\[ u^1 = (a_1, a_2, \ldots, a_{n-1}, 1 - \sum_{i \neq n} a_i) \]
\[ u^2 = (a_1, a_2, \ldots, 1 - \sum_{i \neq n-1} a_i, a_n) \]
\[ \vdots \]
\[ u^n = (1 - \sum_{i \neq 1} a_i, a_2, \ldots, a_{n-1}, a_n) \]

Notice that the condition \( \sum_{i=1}^{n} a_i < 1 \) guarantees that these extreme points satisfy the conditions of lemma 1.

**Theorem 1**: Consider the extreme points \( u^1, u^2, \ldots, u^n \) of the polytope \( Q(a) \) and the surface \( \prod_{i=1}^{n} u_i^{a_i} = \beta \) going through these \( n \) points. The coefficients \( a_i \) defining this surface are strictly positive.

**Proof**: Through a logarithmic transformation the surface \( \prod_{i=1}^{n} u_i^{a_i} = \beta \) is mapped into a hyperplane.
\[
\sum_{i=1}^{n} \alpha_i v_i = \log \beta
\]
going through the points \(v^1, v^2, \ldots, v^n\) with \(v^k = (\log a_1, \log a_2, \ldots, \log (1 - \sum_{i \neq k}^{n} a_i), \ldots, \log a_n)\).

Since \(1 - \sum_{i=1}^{n} a_i > 0\)

it follows that \(1 - \sum_{i \neq k}^{n} a_i > a_k\), \(\forall k\)

and \(\log(1 - \sum_{i \neq k}^{n} a_i) > \log a_k\), \(\forall k\)

This inequality states that in the translated coordinate system \(y = \log a\)
a point \(v^k\) lies on the positive \(k\)-axis. Since this is true for all \(k\),
the coefficients \(\alpha_i\) must be strictly positive.

**Theorem 2:** Given a sequence \((u^1, k, u^2, k, \ldots, u^n, k)\) where each set of points
in the sequence are the extreme points of a polyhedron \(Q(a^k)\) as defined
before. Then

\[
\lim_{k \to \infty} \lambda^k_{\sup} = 0 \text{ if } (u^1, k, u^2, k, \ldots, u^n, k) \to (\bar{u}, \bar{u}, \ldots, \bar{u})
\]

**Proof:** Consider a sequence \(\{u^{0, k}\}_{k=1}^{\infty}\) of points belonging to the surface
\[\prod_{i=1}^{n} u_i^{\alpha_i} = \beta\] through the \(n\) points \((u^1, k, u^2, k, \ldots, u^n, k)\) and assume that \(u^{0, k} \to \bar{u}\).

Because of the convexity of the set \(U \equiv \{u > 0 | \prod_{i=1}^{n} u_i^{\alpha_i} = \beta\}\) when \(\alpha_i > 0\),
a any tangent plane to the surface at \(u^{0, k}\) is a supporting hyperplane of the corresponding set \(U\).
The theorem will be proved if we can show that the tangent plane at \( u^{0,k} \) goes to \( \{ u \mid \sum_{i=1}^{n} u_i = 1 \} \) when \( k \) goes to \( \infty \).

Given the approximating surface \( \prod_{i=1}^{n} u_i^{\alpha_i} = \beta \) through \( n \) points of the simplex satisfying the conditions of lemma 1, the tangent plane at a point \( \tilde{u} \) is given by:

\[
\sum_{i=1}^{n} \left( \alpha_i \tilde{u}_i^{\alpha_i-1} \prod_{j \neq i} \tilde{u}_j^{\alpha_j} \right) u_i = \beta \sum_{i=1}^{n} \alpha_i u_i
\]

or

\[
\sum_{i=1}^{n} \alpha_i \tilde{u}_i^{\alpha_i-1} u_i = \sum_{i=1}^{n} \alpha_i u_i
\]

If we denote by \( a^k > 0 \) the vector of constants defining the polyhedron \( Q(a^k) \), and by \( \alpha_i, i = 1, \ldots, n \) the exponents in the approximating surface through the extreme points of \( Q(a^k) \), it can easily be derived that the following relations must hold:

\[
\frac{a^k_i}{a^k_j} = \frac{\log \left[ \frac{(1 - \sum_{1 \neq j} a^k_i)/a^k_j}{1 - \sum_{1 \neq i} a^k_i/a^k_i} \right]}{1 \neq j}, \text{ for all } i, j \text{ with } i \neq j.
\]

Clearly the only way in which the sequence \( (u^1, k, \ldots, u^n, k) \) can go to \((\bar{u}, \bar{u}, \ldots, \bar{u})\) is having \( a^k \to \bar{u} \) and, as a result \( \sum_{i=1}^{n} a^k_i + 1 \). It is then easy to show that

\[
\frac{\bar{u}_i}{\alpha_i} = \lim_{k \to \infty} \frac{\alpha_i}{\alpha_j} = \frac{u_i}{u_j}, \text{ for all } i \neq j
\]
Consider program (R) as defined in section II. Because of the assumption that $0 < l_j \leq t_j \leq u_j < +\infty$, every term $u_i$ is bounded below by a positive number, say

$$0 < e_i \leq u_i, \text{ i.e. } [k], k = 1, 2, \ldots, p + r.$$ 

The number of terms in a constraint $k$ will be denoted by $T_k$ where of course $T_k = n_k - m_k + 1$, $k = 1, \ldots, r + p$. Moreover we define

$$C \equiv \{ \mu \in \mathbb{R}^{k=1} | e_k(t) \in \sum u_i(t) \leq 1, k = r+1, \ldots, r+p; t > 0 \}$$

Each program to be solved in the branch and bound procedure can be characterized by a set of $r$ vectors:

$$a_{nk} = (a_{k1}, a_{k2}, \ldots, a_{kr}, t_k), k=1, \ldots, r, \text{ with } a_{kj} > 0 \text{ and } \sum_{j=1}^{r} a_{kj} < 1.$$ 

The vector $a_{nk}$ is associated with reversed constraint $k$ and defines a polyhedron $Q(a_{nk})$ in the simplex corresponding to the $k^{th}$ reversed constraint,
as explained above. An approximation \[ \prod_{j}^{k} u_{j} \geq \beta_{k}, \quad k = 1, \ldots, r \] is then constructed through the \( T_{k} \) extreme points of \( Q(\alpha_{k}) \). The program to be solved at each step in the algorithm can then be written as

\[
\inf_{\substack{\alpha \in C}} g_{0}(\tau)
\]

Subject to \( \tau \in C \)

\[
\prod_{j}^{k} u_{j} \geq \beta_{k}, \quad k = 1, \ldots, r
\]

\[
L_{j} \leq t_{j} \leq U_{j}
\]

Every such subprogram will be associated with a node in a branch and bound tree and we now explain how subsequent nodes are generated. At every iteration a pendant node and an index \( k \in \{1, \ldots, r\} \) (according to rules described below) will be singled out. Assume that the node is characterized by the vectors \( \alpha_{1}, \alpha_{2}, \ldots, \alpha_{k} \). A point \( \alpha \in \mathbb{R}^{n} \) is selected such that \( \alpha \in \text{int} \left[ Q(\alpha_{k}) \right] \) in a way to be described below. A total of \( T_{k} \) successors of the current node are then obtained, where the \( i \)th successor is characterized by the vectors:

\[
(a_{1}, a_{2}, \ldots, a_{k-1}, a_{k+i}, \ldots, a_{n}) \quad \text{for} \quad i = 1, 2, \ldots, T_{k}
\]

where \( a_{k} = (a_{k,1}, a_{k,2}, \ldots, a_{k,i-1}, \hat{a}_{k,i}, a_{k,i+1}, \ldots, a_{k,n}) \).

An example for the case \( T = 3 \) is given in Figure 4 where the polyhedra \( Q(\alpha_{1}) \), \( Q(\alpha_{2}) \) and \( Q(\alpha_{3}) \) are visualized.

We are now able to proceed with the first of two algorithms (the relaxed algorithm). It is simpler than the "strict algorithm" with respect to
computational effort involved but has weaker convergence properties.

Relaxed Algorithm

In this algorithm a sequence of points \( \{u^q\} \) will be constructed with the property that the final point is an optimal solution of program (R) when the procedure stops after a finite number of steps. If an infinite sequence is generated then the strongest result we are able to show is the existence of a subsequence converging to an optimal solution of (R).
Initiation Step: Solve program $P^1$ characterized by the vectors

$$a^1_1 = (\varepsilon_1, 1, \varepsilon_1, 2, \ldots, \varepsilon_1, T_1)$$

$$\vdots$$

$$a^1_r = (\varepsilon_r, 1, \varepsilon_r, 2, \ldots, \varepsilon_r, T_r)$$

with $\varepsilon_k, j = \varepsilon_{m_k} + j - 1$ and $\varepsilon_{m_k} + j - 1$ as defined before. Let the solution be denoted by $u^h = \sum_{k=1}^{r+p} u^h_k$ and the objective function value by $\mu^1$.

$p^1$ corresponds to the starting node of the branch and bound tree.

Iteration $q$

(i) Select a pendant node $P^h$ in the tree such that

$$u^h = \min \{ u^i | u^i = \min (P^i), i \in Z(q) \}$$

where $Z(q)$ is the set of pendant nodes at iteration $q$. Let the solution vector of $P^h$ be $u^h_q$. Define $u^q = u^h_q$ and $\mu^q = \mu^h$.

(ii) If $u^h_q$ is feasible for program $(R)$, it is also optimal. (STOP). Otherwise, go to (iii)

(iii) Select an index $k(q) \in \{1, 2, \ldots, r\}$ such that

$$1 - \sum_{k=1}^{r} u^h_k = \max_{[k(q) \in Z(q) \cap [k]} [1 - \sum_{k=1}^{r} u^h_k]$$

(iv) Let $P^h$ be characterized by $(a^h_1, \ldots, a^h_r)$ and define the set
The condition on $K$ guarantees that $S(q) \neq \emptyset$, as can easily be checked.

(v) Select any point $\frac{q}{n_k(q)} \in S(q)$ (notice that $\frac{q}{n_k(q)} \in \text{int} [Q(a_k^h)]$) and generate $T_k(q)$ successors of $P^h$ as explained above, the $i$th successor characterized by

$$(a^1, a^2, \ldots, a^h_{k(q)-1}, a^i_k(q), a^i_k(q)+1, \ldots, a^h_k(q))$$

with

$$a^i_k(q) = (a^1_k(q), \ldots, a^h_k(q), i-1, a^i_k(q), i, a^i_k(q)+1, \ldots, a^h_k(q), T_k(q))$$

(vi) Solve all successors of $P^h$, set $q = q + 1$ and go to (i).

The strict version of the algorithm, as described below, guarantees an optimal solution if the method terminates in a finite number of steps or an infinite sequence $\{u^q\}$ such that every converging subsequence converges to an optimal solution of program (R).
**Strict Algorithm**

**Initiation:** see relaxed algorithm

**Iteration q**

(i) Select pendant node $P^h$ in the tree with the smallest objective function value. Let $P^h$ be characterized by $(a^h_1, \ldots, a^h_n)$ and denote its solution by $u^h_\omega$ and the objective function value $\mu^h$.

(ii) and (iii) See "relaxed algorithm".

(iv) If the polyhedron $Q(a^h_{k(q)})$ has not been partitioned at any previous iteration $q'$ in the algorithm; go to (v). Otherwise, go to (vi).

(v) Assume that, so far, $\{u^j_\omega, \mu^j_\omega\}_{j=1}^{1+1}$ has been generated. Set

$\begin{align*}
    u^1_\omega &= u^h_\omega \\
    \mu^1_\omega &= \mu^h
\end{align*}$

and perform steps (iv), (v) and (vi) of the relaxed algorithm.

(vi) Perform steps (v) and (vi) of the relaxed algorithm but take

$\begin{align*}
    a^k_\omega(q) &= a^k_\omega(q')
\end{align*}$

in step (v).

**Remark 1:** Every subproblem to be solved in the algorithms is a prototype geometric program. Hence it can be transformed into a convex program.

**Remark 2:** The important difference between the two algorithms is that, in the relaxed algorithm, the solution of every selected pendant node in step (ii) is incorporated in the sequence $\{u^q_\omega\}$, whereas in the strict algorithm this is only the case when the subsimplex $Q(a^h_{k(q)})$ defining the approximation
to reversed constraint k(q) has not been partitioned at any previous iteration.

V. CONVERGENCE RESULTS

We first prove convergence of the strict algorithm.

Theorem 3: When the strict algorithm stops in a finite number of steps, the last point generated is optimal for (R). If a sequence \( \{u_q^q\} \) is generated then every converging subsequence of \( \{u_q^q\} \) converges to an optimal solution of (R).

Proof:

It is immediately clear that the last point generated in the sequence is optimal when the procedure stops after a finite number of iterations. Therefore, assume that a sequence \( \{u_q^q\} \) is generated and let \( \{u_i^i\} \) be a subsequence converging to \( u^\infty \) (there exists at least one by compactness). By contradiction, assume that \( u^\infty \) is not an optimal solution of (R). Because of step (i) it must be true that, for some \( k \in \{1, 2, \ldots, r\} \)

\[
\sum_{[k]} u^\infty_{x_j} < 1
\]

Let \( \{P_i^i\} \) be the sequence of programs(nodes) corresponding to \( \{u_i^i\} \). Constraint k has been selected for partitioning either a finite or an infinite number of times in this sequence (in step (iii)). First, assume it has been selected an infinite number of times. Consider a subsequence \( \{u_s^s\} \) from \( \{u_i^i\} \) such that constraint k was selected for every s and such that \( 1 - \sum_{[k]} a_s^i \) converges to \( 1 - \sum_{[k]} a_i^i \). Since \( \sum_{[k]} u^\infty_{x_i} < 1 \),
it follows that $1 - \sum_{i=1}^{k} a_i^\infty > 0$. Hence, for $s$ large enough, there exists $(u_m, \ldots, u_n)$ such that

$$\sum_{i=1}^{k} u_i = 1$$

and

$$u_i \geq a_i^s + \frac{1}{2} \frac{(1 - \sum_{i=1}^{k} a_i^\infty)(1 - \sum_{i=1}^{k} u_i^\infty)}{|k| l k U^0}, \forall i \in [k]$$

where $U^0$ is some upper bound for $\sum_{i=1}^{k} u_i$.

Now, we have that

$$\delta = \frac{1}{2kU^0} (1 - \sum_{i=1}^{k} a_i^\infty)(1 - \sum_{i=1}^{k} u_i^\infty) > 0$$

This means that, for $s$ large enough the $i$th coordinate of the vector $a_i^k(s)$ ($k(s) = k, \forall s$) characterizing the $i$th successor of $P^s$ is increased by at least $\delta > 0$ (see step v). Since a point $u_x^s$ is only in the sequence $u_x^\infty$ when $Q(a_k^s)$ is partitioned for the first time, this contradicts $\sum_{i=1}^{k} u_i^\infty < 1$.

Secondly, assume a constraint $k'$ has been selected a finite number of times and let $\sum_{i=1}^{k'} u_i^\infty < 1$. This is clearly impossible because of selection rule (iii) and the previous result for constraints where simplices were partitioned infinitely often. Q.E.D.

**Theorem 4**: When the relaxed algorithm terminates after a finite number of steps, then the last point generated in the sequence $\{u_q^s\}$ is optimal for (R). If an infinite sequence $\{u_q^s\}^\infty_{s=1}$ is generated, then there exists at least one subsequence converging to an optimal solution of (R).

**Proof**: The proof is immediate when the algorithm stops after a finite number of steps.
If an infinite sequence is generated, then there exists at least one infinite path in the branch and bound tree starting from \( P_1 \). An analogous reasoning as in the previous convergence proof can be applied to this path.

VI. SPECIAL CASES

1. Barycentric Method.
   
The selection of a point \( \hat{a}_{k}(q) \) in step (v) has not been uniquely defined. One possibility is to take the barycenter of the polyhedron \( Q(a_{k}(q)) \) for which it can be checked immediately that it belongs to \( S(q) \) (see fig. 5 for an illustration when \( T_k(q) = 2 \)).

2. Combination with Passy's Algorithm [7]

If we add the constraints

\[ u_{[k]} \geq a_{[k]^h}, \quad k = 1, 2, \ldots, r \quad \text{and} \quad u_{[k]} = (u_{m_{k}}, \ldots, u_{n_{k}}) \]

to subprogram \( P^h \) characterized by the vector \( (a_{1}^h, \ldots, a_{r}^h) \), we have essentially a combination between Passy's algorithm and ours (it must be pointed out that adding these constraints is redundant when \( T_k = 2 \), but not in higher dimensions, as in clear from the first part of this paper). The point \( \hat{a}_{k}(q) \) in step (v) is selected by Passy as (see fig. 5).

\[
\hat{a}_{k}(q), j = \frac{u_{j}^h}{\sum_{j} u_{j}^h} , \quad j \in [k(q)]
\]

For \( \hat{a}_{k}(q) \in S(q) \) to be true, it must hold that
This inequality holds whenever

\[
\frac{u_j^h}{a_j} \geq \frac{1 - \sum_{k(q)} a_j^h}{1 - \sum_{k(q)} u_j^h} + \frac{1 - \sum_{k(q)} u_j^h}{1 - \sum_{k(q)} a_j^h} , \quad j \in [k(q)]
\]

or whenever

\[
\frac{u_j^h}{a_j} \geq \frac{1 - \sum_{k(q)} a_j^h}{\sum_{k(q)} a_j^h} \frac{1 - \sum_{k(q)} u_j^h}{\sum_{k(q)} u_j^h} \frac{1 - \sum_{k(q)} u_j^h}{1 - \sum_{k(q)} a_j^h} < 1 \quad \text{since} \quad \frac{u_j^h}{a_j} > 1
\]

or whenever

\[
K \geq \max_{k=1, \ldots, r} \left\{ \frac{1 - \sum_{[k]} \varepsilon_{i}}{\min_{i \in [k]} \varepsilon_{i}} \right\}
\]

Hence, if we redefine K in the algorithm as constrained by

\[
K \geq \max_{k=1, \ldots, r} \left\{ \frac{T_k}{\sum_{[k]} \varepsilon_{i}} , \frac{1 - \sum_{[k]} \varepsilon_{i}}{\min_{[k]} \varepsilon_{i}} \right\}
\]

the convergence proof holds for this modified algorithm.

Figure 5.
VII. EXAMPLE

We solve the well-known "Gravel-Box" problem [4] which was also solved in Passy [7].

\[
\min 40 t_1^{-1} t_2 t_3^{-1} + 20 t_2 t_3 + 40 t_1 t_2 + 10 t_1 t_3
\]

Subject to \[ t_1 + t_2 + t_3 \geq 8 \]

The global minimum is known to have an objective function value of 121.41 [7].

We first solved the problem by transforming it into a problem of the form (T) with only two term reversed constraints.

\[
t_1 + t_4 \geq 8 \quad \text{or} \quad .125 t_1 + .125 t_4 \geq 1
\]
\[
t_2 + t_3 \geq t_4
\]
\[
t_2 t_4^{-1} + t_3 t_4^{-1} \geq 1
\]

The relaxed algorithm with \( \epsilon_i = .01 \), all \( i \), and selecting the barycenter in step (v) resulted in the branch and bound tree of fig. 6. The nodes are numbered in the order that the subprograms were solved. To reach a lower bound of 121 we had to solve 19 subproblems. \( P_{18} \) however has a value of 121.47 and is very close to feasibility (first constraint is satisfied and the second to within .003).

The same problem was solved in its original form and using approximations in dimension three. Here we had to solve 19 subproblems to attain a lower bound of 120.45, slightly worse than in the previous case (fig.7). However in this case several feasible solutions were generated providing good upper bounds (best upper bound is given by program \( P_7 : 122.17 \)). Nodes
Fig. 6.

\[ P_1 (100) \]
\[ P_2 (100) \]
\[ P_3 (11.6) \]
\[ P_4 (102.0) \]
\[ P_5 (140.0) \]
\[ P_6 (105.9) \]
\[ P_7 (136.3) \]
\[ P_8 (115.0) \]
\[ P_9 (166.0) \]
\[ P_{10} (117.4) \]
\[ P_{11} (132.7) \]
\[ P_{12} (118.96) \]
\[ P_{13} (127.0) \]
\[ P_{14} (119.9) \]
\[ P_{15} (121.01) \]
\[ P_{16} (128.3) \]
\[ P_{17} (133.3) \]

\[ P_{18} : t_1 = .52 \]
\[ t_2 = .25 \]
\[ t_3 = 7.23 \]
\[ t_4 = 7.50 \]

\[ P_{19} : t_1 = .54 \]
\[ t_2 = .25 \]
\[ t_3 = 7.09 \]
\[ t_4 = 7.42 \]
$t_1 = 0.53$
$t_2 = 0.26$
$t_3 = 7.00$

Fig. 7.
providing feasible solutions are indicated by a cross. Passy's method required the solution of 50 subprograms to attain the lower bound 119.54. To attain a similar bound we have to solve 15 subprograms in the first case and 16 in the second. (Lower bounds of 119.6).

VIII. CONCLUSIONS

Several branch and bound schemes have been suggested for solving signomial geometric programs. The basic problem was to partition a simplex such that the approximation $\prod u_i^{\alpha_i} \geq \beta_k$ to a part of the feasible region of reversed constraint $\mathbf{x}$ (defined by a subsimplex) was such that convergence of the algorithm could be shown. In fact, we could have started the paper with section IV, immediately introducing the partitioning proposed there. However we think that, in such a case, the question would arise as to why exactly that partitioning was used and not a more arbitrary one. The answer of course was implied by the conditions which were imposed on the selection of the subsimplices in order for the approximation to be "well-behaved". Of course other partitionings than the one proposed may be possible and these would be especially useful if a simplex were subdivided into non-overlapping regions. It must also be pointed out that the algorithms and several of the proofs could have been much simplified under less general assumptions, i.e. by always working with problems having not more than two terms in a reversed constraint (this implies no loss of generality) or by always picking the barycenter to generate the subsimplices.
REFERENCES


