AVENUES OF GEOMETRIC PROGRAMMING*

T.R. JEFFERSON AND C.H. SCOTT
THE UNIVERSITY OF NEW SOUTH WALES, AUSTRALIA

Summary

Since its initial appearance more than a decade ago, geometric programming has proved itself to be a powerful means of solving many optimization problems, particularly those pertaining to engineering design. Recently the theory has been generalised to provide a structure for the analysis of any convex programming problem. Here we trace the development of geometric programming from its initial use as a trick for solving certain cost optimization problems to its current status as a fully fledged branch of optimization theory. Examples drawn from inventory theory reinforce and motivate this development.

1. Introduction

In the early sixties, Clarence Zener was a Director of Science at Westinghouse Research Laboratories at Pittsburgh. In such a position, a physicist becomes involved with cost modelling. The difference, in Zener's case, was that he observed a curious mathematical phenomena with regard to cost models made up as a sum of component costs [47,48]. He was able to optimize certain such models by mere inspection of the exponents of the design variables (Section 2). Thus was born a new and exciting approach to mathematical programming which has henceforth been termed geometric programming.

Initial research concentrated on posynomial (polynomial with positive coefficients) programs where both the objective function and constraints are in posynomial form (Section 3). This research culminated in the seminal book entitled "Geometric Programming" by Duffin, Peterson and Zener [13]. Subsequent work developed in two significantly different directions: Signomial Programming (Section 4) and Generalised Geometric Programming (Section 5) [33].

*Manuscript submitted October 28, 1977
Signomial Programming, first developed by Passy and Wilde [32], sought to relax the restriction of the functions treated by Geometric Programming to the form of polynomials. With the relaxation to polynomials one loses convexity and hence one can only hope for a local minimum, not a global minimum. Generalised Geometric Programming proceeds to generalise the type of function considered from posynomial to convex. This generalisation does maintain the convexity of the problem and hence one is able to find the global solution to the mathematical program.

The relationship among the various developments may outwardly seem tenuous. This is because Geometric Programming should not be thought of as a class of mathematical programs but as a method of analysis of mathematical programs. In this respect, geometric programming has much in common with dynamic programming. Dynamic Programming makes use of recursive functions. Geometric Programming makes use of linearity, separability, convexity, and duality.

(i) **Linearity**

Most mathematical programming is carried out over linear vector spaces. The usual choice is Euclidean n-dimensional space ($\mathbb{E}^n$). Here a vector is represented by an n-tuple of real components. Taking any two vectors $x_1$ and $x_2$ belonging to the space, and real scalars $\alpha$ and $\beta$, a linear vector space has the property that the linear combination $\alpha x_1 + \beta x_2$ belongs to the space. In the context of functions, a function $f(x)$ is said to be linear if for any scalars $\alpha$ and $\beta$, $f(\alpha x + \beta y) = \alpha f(x) + \beta f(y)$. Linear functions have many important properties. They are both convex and concave. Further, a first order Taylor series expansion approximates the linear function exactly. The value of linearity to mathematical programming is best exemplified by the fruitful area of linear programming.
(ii) **Separability**

A function \( f(x_1, x_2, \ldots, x_n) \) is separable if it can be written as \( f(x_1, x_2, \ldots, x_n) = f_1(x_1) + f_2(x_2) + \ldots + f_n(x_n) \). This property is important as it allows one to treat an n-dimensional function as n functions of 1-dimension. This is much simpler from a computational point of view.

(iii) **Convexity**

A set \( A \) is convex if for any \( x_1, x_2 \in A \) and \( 0 \leq \lambda \leq 1 \), \( \lambda x_1 + (1-\lambda) x_2 \in A \). To extend the concept of a function \( f(x) \) defined for \( x \in C \), a convex set, we require the following definition. The epigraph of a function \( f \) defined on \( C \) is the set \( A \) defined by: \( A = \{(a, x) | a \geq f(x), x \in C\} \). \( f(x) \) is a convex function iff \( A \) is a convex set. Convex functions have a number of powerful properties. The most important for mathematical programs with a convex objective function and a convex constraint set is that any local minimum is also a global minimum. Further properties of convexity will be given in Section 5.

(iv) **Duality**

For a linear space \( X = \mathbb{F}_n \), linear functions may be written in the form \( <a, x> = \sum_{i=1}^{n} a_i x_i \), where \( a \) is a vector parameter. The set of all linear functions (parameterised by the vector \( a \)) is a space and is called the dual space \( (X^*) \). The duality arises from the fact that \( <a, x> \) is symmetric in \( a \) and \( x \). For finite dimensional spaces, the set of linear functions on \( X^* \) turns out to be the original space \( X \). Hence there is a symmetric relationship between \( X \) and \( X^* \).

Of importance to mathematical programming is the pairing of sub-spaces in primal and dual spaces with respect to the inner product. Let \( X \) be a subspace of \( X \). Then \( X^\perp = \{y | <y, x> = 0, \forall x \in X\} \) is called the orthogonal complementary subspace of \( X \). Any point in \( X^\perp \) is orthogonal to any point in \( X \) by construction. If \( y_1, y_2 \in X^\perp \) and \( \alpha, \beta \) are
scalars, it is easy to see that $\alpha X_1 + \beta X_2 \in X$.

The importance of the concepts of linearity, separability, convexity and duality in the development of geometric programming will remain obscure in Sections 2, 3 and 4. However, their paramount role will become clear in Section 5 when we introduce Generalised Geometric Programming. Section 6 uses this generalised theory to re-derive the theory of posynomial programs, and the central role played by these concepts will be further clarified. Throughout this survey, theory will be reinforced by examples drawn, in the main, from the area of inventory theory. The examples presented are relatively simple in order to highlight the use of geometric programming theory.

2. Unconstrained Posynomial Programming

Here we consider the following nonlinear problem:

$$\text{Minimize } g(t) = \sum_{i=1}^{n} u_i$$ (1)

where $u_i$, $i=1,\ldots,n$, are posynomials defined by

$$u_i = \sum_{j=1}^{m} c_i^j t_j^{a_{ij}}$$ (2)

$c_i > 0$, $\forall i$, $t_j > 0$, $\forall j$, and $a_{ij}$ are arbitrary real constants; $t = (t_1, \ldots, t_m)^T$ and $T$ denotes a transpose. In the usual situation, $g(t)$ represents a composite cost made up of component costs $u_i$, $i=1,\ldots,n$. We term this the primal problem.

In principle, the primal problem could be solved by the methods of differential calculus. This will give rise to a set of nonlinear equations which, in general, are difficult to solve. Hence, in practice, one would invariably resort to a numerical solution by an iterative descent-type method. Here, we propose to construct a geometric programming dual program to the primal. In certain cases, this admits a trivial solution to the problem. Further, this dual
problem has an interesting and informative practical interpretation.

A central idea here is the arithmetic-geometric mean inequality, henceforth termed the geometric inequality. In fact, the name "Geometric Programming" comes from the importance of this inequality to the original theory. It states that

\[ \sum_{i=1}^{n} \delta_i v_i \geq \prod_{i=1}^{n} (v_i) \delta_i \]  

where \( \sum_{i=1}^{n} \delta_i = 1, \delta_i \geq 0, v_i, v_i > 0, \forall i \). Equality is attained when \( v_1 = v_2 = \ldots = v_n \). It is convenient to set \( v_i = u_i / \delta_i, \forall i \) in equation (3) to obtain

\[ \sum_{i=1}^{n} u_i \geq \prod_{i=1}^{n} (u_i / \delta_i) \delta_i \]  

In this form, we can use the geometric inequality to obtain a lower bound on our primal optimization problem. Hence, substituting equation (2) into (1) and using (4), we have that

\[ g(t) \geq \prod_{i=1}^{n} (c_i / \delta_i) \delta_i \prod_{j=1}^{m} t_j^{a_{ij} \delta_i} \]  

If we choose the weights, \( \delta_i, \forall i \), such that

\[ \sum_{i=1}^{n} a_{ij} \delta_i = 0, \quad j=1,\ldots,m \]  

then the variables \( t_j, j=1,\ldots,m \) on the right hand side of inequality (5) may be eliminated. In this case, we have that

\[ g(t) \geq \prod_{i=1}^{n} (c_i / \delta_i) \delta_i = v(\delta) \]  

where \( v(\delta) \) is the dual function which gives a lower bound on the minimum of \( g(t) \). Equation (7) implies that

\[ \min g(t) \geq \max v(\delta) \]
under conditions $t > 0$, $\delta > 0$, $\sum_{i=1}^{n} \delta_i = 1$ and equation (6).

It may be shown that there exists an optimal $t^*$ (in the sense of minimizing $g(t)$) and an optimal $\delta^*$ (in the sense of maximizing $v(\delta)$) to satisfy inequality (8) at equality. In this case the relation between $t^*$ and $\delta^*$ is given by

$$\delta_i^* = \frac{u_i(t^*)}{g(t^*)}$$

(9)

Hence the following dual program to the original primal program may be constructed:

Maximize $v(\delta) = \prod_{i=1}^{n} \frac{c_i}{\delta_i}$

subject to the normalisation condition

$$\sum_{i=1}^{n} \delta_i = 1, \delta_i \geq 0$$

(11)

and the orthogonality condition

$$\sum_{i=1}^{n} a_{ij} \delta_i = 0, j=1,\ldots,m$$

(12)

At optimality $g(t^*) = v(\delta^*)$.

From equation (9), we see that we can interpret the dual variables at optimality, $\delta_i^*$, $\forall i$, as the relative contribution of each component cost $u_i$ to the composite cost $g(t)$. Further, we obtain the following relationship between the primal and dual variables

$$\ln \left( \frac{\delta_i^* v(\delta^*)}{c_i} \right) = \sum_{j=1}^{m} a_{ij} \log t_j^*$$

(13)

This is a system of linear equations in $\log t_j$, $j=1,\ldots,m$ which are readily solvable once the dual program has been solved.

We note that the dual program maximizes a nonlinear function subject to linear equality constraints. A dual variable $\delta_i$ is associated with each term $i=1,\ldots,n$ in the primal formulation. In the case that the number of terms in
the primal $n$ is equal to the number of variables in the primal $m$ plus one, i.e., $n=m+1$, the linear constraints admit a unique solution $\xi^*$ and the optimization problem in the dual is trivial. The quantity $n-(m+1)$ is termed the degree of difficulty of a geometric program and is, in some sense, a measure of the computational complexity of the dual program.

Example

We consider the well known "Economic Lot Size" problem from inventory theory. Items are withdrawn continuously from inventory at a known constant rate $a$. Items are ordered in equal numbers, $Q$, at a time and production is instantaneous. The problem is to determine how often to make a production run and how much to order, $Q$, each time to minimize the cost, $C$, per unit time, where $C = aK/Q + hQ/2 + ac$, and $K$ is the fixed set-up cost, $h$ is the inventory holding cost per item per unit time, and $c$ is the variable production cost per item. Neglecting the constant part, $ac$, we require to

$$\text{minimize } C_1 = aK/Q + hQ/2$$

This is an unconstrained posynomial program. The corresponding dual program, from equations (10), (11) and (12), is to maximize

$$(aK/\delta_1) \delta_1 (h/2\delta_2) \delta_2$$

subject to $\delta_1 + \delta_2 = 1$ (normality)

$$-\delta_1 + \delta_2 = 0$$

$$\delta_1 \geq 0, \delta_2 > 0$$

Here the constraint equations can be uniquely solved to yield $\delta_1^* = \delta_2^* = \frac{1}{2}$ and there is no maximization problem. We have a program with zero degree of difficulty. The interpretation of this result is that at optimality, the set-up
cost per unit time and the holding cost per unit time contribute equal amounts to the optimal cost, i.e., the optimal distribution of cost is an invariant with respect to the cost coefficients. Hence, from equations (10) and (13), the optimal cost is $\sqrt{2aK/h}$, and the optimal order quantity is $J^{TaTk}/h$.

Suppose now that the production process is a lengthy one so that the assumption of instantaneous availability is no longer acceptable. Furthermore, the size of the lot determines the length of the production process, and this in turn determines the in-process inventory holding cost. In this case a modified economic lot size model could be to minimize $C_1 = aK/Q + hQ/2 + IQT/2$, where $I$ is in-process inventory holding cost per unit time, and $T$ is the length of the production process, given by $T(Q) = nQ + m$, where $n$ and $m$ are empirically determined constants. In this case, we have the unconstrained posynomial program:

$$\text{Minimize } C_1 = aK/Q + Q(h/2+Im/2) + In Q^2/2$$

The corresponding dual program is given by

$$\text{Maximize } (aK/\delta_1) \delta_1 ((h+Im)/2 \delta_2) \delta_2 (m/2\delta_3)^2 \delta_3$$

subject to

$$\delta_1 + \delta_2 + \delta_3 = 1$$

$$-\delta_1 + \delta_2 + 2\delta_3 = 0$$

$$\delta_1 > 0, \delta_2 > 0, \delta_3 > 0$$

Here we have two equations and three unknowns; in effect, a degree of difficulty of one. One could proceed to solve this problem as an unconstrained maximization problem in one variable or as a constrained maximization. However, insight may be gained without going into this detail. Manipulation of the constraint equations gives the following inequalities: $0.5 \leq \delta_1 \leq 1.0$, and $0.33 \leq \delta_1 \leq 0.66$, which imply that at optimality, the set-up cost makes a contribution
between 50% and 66% to the optimal composite cost. Similar insights may be obtained from the other dual variables.

3. Constrained Posynomial Programming

Here we consider the minimization of a posynomial form subject to constraints which are also posynomials. This is the primal formulation.

Minimize \( g_0(t) \) subject to \( g_k(t) \leq 1, \ k=1, \ldots, p \) (14, 15)

\[ t_j > 0, \ j=1, \ldots, m \] (16)

where \( g_k(t) = \sum_{i \in [k]} c_i \prod_{j=1}^m a_{ij} t_j \), \( k=0, \ldots, p \) (17)

\[ [k] = \{m_k, m_k+1, \ldots, n_k\}, \ k=0, \ldots, p \] (18)

and \( m_0=1, m_1=n_0+1, \ldots, m_p=n_p-1, n_p=n \).

\( \{k\}, k=0,1, \ldots, p \) is a sequential partition of the integers 1 to n. As before \( a_{ij} \) are arbitrary real exponents and the coefficients \( c_i \) are positive.

In order to handle constraints, we need a generalisation of the geometric inequality in which the \( \delta_i \)'s are not normalised. We let

\[ \lambda_k = \sum_{i \in [k]} \delta_i \] (19)

for a particular constraint \( k \). Hence the geometric inequality, equation (4) may be written in a convenient form as

\[ (\sum_{i \in [k]} u_i)^{\lambda_k} \geq \prod_{i \in [k]} (u_i/\delta_i)^{\delta_i} \lambda_k \] (20)

Combining equations (15), (17) and (20) we have that

\[ 1 \geq g_k(t)^{\lambda_k} \geq \prod_{i \in [k]} (c_i \prod_{j=1}^m a_{ij}/\delta_i)^{\delta_i} \lambda_k \] (21)

for any constraint \( k=1, \ldots, p \). For the objective function, we normalise the \( \delta_i, i \in [0] \). Hence
\[ g_0(t) \geq \prod_{i \in [0]} (c_i \prod_{j=1}^{n} a_{ij}/\delta_i) \delta_i \]  
\[ \text{and} \quad \sum_{i \in [0]} \delta_i = 1 \]  

Combining results (21) and (22), we obtain the inequality

\[ g_0(t) \geq \prod_{i=1}^{n} (c_i/\delta_i) \delta_i p \prod_{k=1}^{\lambda_k} \]  

Hence, using the same lines of reasoning as for the unconstrained problem, the dual problem is given by:

\[ \text{Maximize } v(\delta) = \prod_{i=1}^{n} (c_i/\delta_i) \delta_i p \prod_{k=1}^{\lambda_k} \]  

subject to the linear constraints

\[ \sum_{i \in [0]} \delta_i = 1 \quad \text{(normalisation)} \]  
\[ \sum_{i=1}^{n} a_{ij} \delta_i = 0, \quad j=1,\ldots,m \quad \text{(orthogonality)} \]  
\[ \delta_i \geq 0, \quad i=1,\ldots,n \]

We note that the primal problem is a highly nonlinear mathematical program, whereas the dual is the maximization of a concave function subject to linear constraints. Once again, if \( n=m+1 \), the program has zero degree of difficulty, and the dual program is trivial. It may be shown [13] that at optimality the primal variables \( t^* \) and the dual variables \( \delta_0^* \) are related by

\[ c_i \prod_{j=1}^{m} a_{ij} t^{*}_{ij} = \begin{cases} \delta_i^* v(\delta^*), & i \in [0] \\ \delta_i^*/\lambda_k(\delta^*), & i \in [k], \lambda_k(\delta^*)>0, k=1,\ldots,p \end{cases} \]  

Hence \( \delta_i^*, i \in [0] \) gives the relative contribution of each component cost to the composite cost. Also at optimality, the primal and dual objective functions are
equal in value, i.e., \( g_0(t^*) = v(\delta^*) \).

**Example**

We consider an Economic Lot Size model with multi-products and a resource constraint. In this case we have the following constrained posynomial program:

\[
\text{Minimize } \sum_{i=1}^{N} \left( \frac{a_i K_i}{Q_i} + \frac{h_i Q_i}{2} \right)
\]

subject to a resource constraint

\[
\sum_{i=1}^{N} b_i Q_i \leq W
\]

where \( b_i \) and \( W \) are given constants. From equations (25) to (28), we find the corresponding dual program:

\[
\text{Maximize } \prod_{i=1}^{N} \frac{(a_i K_i / \delta_i)}{\delta_i} \prod_{i=N+1}^{2N} \frac{(h_i / 2 \delta_i)}{\delta_i} \prod_{i=2N+1}^{3N} \frac{(b_i / W \delta_i)}{\delta_i \lambda}
\]

subject to

\[
\sum_{i=1}^{\lambda} \delta_i = 1
\]

\[
\lambda = \sum_{i=2N+1}^{3N} \delta_i
\]

\[
-\delta_i + \delta_{N+i} + \delta_{2N+i} = 0, \quad i=1, \ldots, N
\]

\[
\delta_i \geq 0, \quad i=1, \ldots, 3N
\]

**4. Signomial Programming**

While posynomial programming has found application in a variety of areas, many problems of interest have fallen outside the posynomial form. Often it was the positivity condition on the coefficients which was violated. However, the benefits of posynomial programming were too tempting to be passed up. Passy and Wilde [32] introduced a generalisation of posynomial programming, where the coefficients of the terms were not required to be positive. This class of programs is called signomial. In signomial programming, a global minimum cannot be guaranteed. However, the properties of the dual subspace are maintained.
Duffin and Peterson [11,12] have developed methods for
the analysis of signomials and an algorithm for solving them
(at least for a local solution, if not for a global solution).
To use the analysis, signomial programs must first be con­
verted into posynomial programs. Any signomial \( f(t) \) can be
written as a difference of two posynomials, say \( r(t) \) and
\( s(t) \) and hence
\[
\begin{align*}
f(t) &< 1 \iff r(t) - s(t) < 1 \iff r(t) \leq s(t) + 1 \\
f(t) &> 1 \iff r(t) - s(t) > 1 \iff r(t) \geq s(t) + 1 \\
f(t) &\leq 0 \iff r(t) - s(t) \leq 0 \iff r(t) \leq s(t)
\end{align*}
\]
Since \( r(t) \), \( s(t) \) and \( s(t) + 1 \) are all non-negative for
all values of \( t \), in each case a new variable \( \tau \) can be intro­
duced so that
\[
\begin{align*}
r(t) &< \tau \iff s(t) + 1 \iff r(t)/\tau \leq 1 \quad \text{and} \quad 1 \leq s(t)/\tau + 1/\tau \\
r(t) &> \tau \iff s(t) + 1 \iff r(t)/\tau \geq 1 \quad \text{and} \quad 1 \geq s(t)/\tau + 1/\tau \\
r(t) &\leq \tau \iff s(t) \iff r(t)/\tau \leq 1 \quad \text{and} \quad 1 \leq s(t)/\tau
\end{align*}
\]
These transformations on a signomial produce a posynom­
ial program with reversed constraints:
\[
\begin{align*}
\text{Minimize } g_0(t) & \quad (30) \\
\text{subject to } & \quad g_k(t) \leq 1 \quad k=1, \ldots, p \quad (31) \\
& \quad g_k(t) \geq 1 \quad k=p+1, \ldots, r \quad (32) \\
& \quad t > Q
\end{align*}
\]
where the functions \( g_k(t) \) are posynomials defined by
equation (17). The difference in this formulation is the
reversed constraints (32). While the constraints (31) de­
fine a convex region, the reversed constraints make the
region non-convex. The dual to the primal program defined
by (30) to (33) is given by:
Maximize \( v(\delta) = \prod_{i \in [0]} (c_i/\delta_i)^{\delta_i} \prod_{k=1}^{r} (c_i/\delta_i)^{\delta_k} \prod_{k=p+1}^{r} (c_i/\delta_i)^{-\delta_i} \)

subject to same constraints as for posynomials, namely

\[ \Sigma_{i \in [0]} \delta_i = 1 \]  
\[ \lambda_k = \Sigma_{i \in [k]} \delta_i, \quad k=1,\ldots,r \]  
\[ \Sigma_{i=1}^{n} a_{ij} \delta_i = 0, \quad j=1,\ldots,m \]  
\[ \delta_i > 0, \quad i=1,\ldots,n \]

We note that \( \log v(\delta) \) is concave in the variables associated with the regular constraints \( k=1,\ldots,p \) and the objective function. However, it is convex in the variables associated with the reversed constraints \( k=p+1,\ldots,r \). In the posynomial case, the dual program required finding the maximum of a concave function subject to linear constraints. Here we are looking for the maximum of a number of saddle points of a concave-convex function subject to linear constraints. If the program has degree of difficulty of zero, it is possible that a solution can readily be found but in the general case, the program is very difficult to solve. In order to cope with this problem, Duffin and Peterson [11] apply the arithmetic-harmonic mean inequality to convert the reversed constraints into regular constraints. This approximation has the advantage that the linear constraints in the dual are the same for all such approximations. As well, the sequence of all regular posynomial programs generated is monotone decreasing in the value of the objective function. Thus one can only improve any feasible solution.

The arithmetic-harmonic inequality, given parameters \( a_i, i \in [k], \) positive and \( \Sigma_{i \in [k]} a_i = 1, \) is

\[ (\Sigma_{i \in [k]} u_i)^{-1} \leq \prod_{i \in [k]} (a_i/u_i)^{a_i} \leq \Sigma_{i \in [k]} a_i^2/u_i \]
with equality holding when
\[ \alpha_i = u_i / (\sum_{i \in [k]} u_i) \]  
(40)

The means in equation (30) are the harmonic, geometric and arithmetic respectively. Consider any reversed constraint
\[ g_k(t) \geq 1 \text{ or } (g_k(t))^{-1} \leq 1 \]

We set \( g_k(t, \alpha) = \sum_{i \in [k]} \alpha_i^2 / u_i \). This is a posynomial for fixed \( \alpha \). We note that \( g_k(t, \alpha) \leq 1 \) implies that \( g_k(t) \geq 1 \).

Using equation (40) for any \( t \) such that \( g_k(t) \geq 1 \), one can generate an \( \alpha \) such that \( g_k(t, \alpha) < 1 \). Hence, by use of the arithmetic–harmonic inequality, we may reduce a posynomial program with reversed constraints to a regular posynomial program parameterised with respect to \( \alpha \), viz.

Minimize \( g_0(t) \)  
(41)

subject to
\[ g_k(t) \leq 1, \quad k=1, \ldots, p \]  
(42)
\[ g_k(t, \alpha) \leq 1, \quad k=p+1, \ldots, r \]  
(43)
\[ t > 0 \]  
(44)

To obtain a monotone decreasing sequence of programs, we solve the program (41) to (44) for a particular \( \alpha \) and then reset the \( \alpha \)'s using equation (40).

Example

We return to the multiproduct economic lot size problem given in Section 3. Here we add consideration of the distribution effort. Sales are now a function of the allocation of effort to distribution, and we seek to maximize profit. The problem becomes:

Maximize \( \sum_{i=1}^{N} (p_i - c_i) a_i d_i / d_i - D_i - K_i a_i d_i / Q_i - h_i Q_i / 3 \)

subject to \( \sum_{i=1}^{N} b_i Q_i \leq W \)
\[ Q_i > 0, \ D_i > 0, \ \forall i \]

Here \( p_i - c_i \) is the sales price minus the cost for product \( i \), \( a_iD_i \) are the sales generated by a distribution effort of \( D_i \) and \( a_i \) and \( d_i \) are parameters. All other symbols have been defined previously. This is a signomial program and is equivalent to the following program:

Minimize \( 1/P \)

subject to \( \sum_{i=1}^{N} b_i Q_i / W < 1 \)

\[ \sum_{i=1}^{N} (p_i - c_i)a_iD_i - d_i - K_a a_i D_i / Q_i - h_i Q_i / 2 > P \] (45)

\[ Q_i > 0, \ D_i > 0, \ \forall i \]

\( P \) is a new variable indicating profit. Taking the negative terms in equation (45) to the right hand side and inserting a new variable \( R \) between the sides of the inequality as in the theory, we obtain the program:

Minimize \( 1/P \)

subject to \( \sum_{i=1}^{N} b_i Q_i / W < 1 \)

\[ P/R + \sum_{i=1}^{d_i} (D_i/R + K_a a_i D_i / (Q_i R) + h_i Q_i / (2R)) < 1 \]

\[ \sum_{i=1}^{(p_i - c_i)a_i D_i} d_i / R > 1 \] (46)

\[ Q_i > 0, \ D_i > 0, \ \forall i \]

Our original signomial program is now in the form of a posynomial program with one reversed constraint (46). This may be harmonised to become

\[ \sum_{i=1}^{(p_i - c_i)a_i D_i} d_i / R < 1 \] (47)

where the \( a_i > 0 \) (and \( \sum a_i = 1 \)) are parameters to be varied. To find a local solution, we take a feasible set of
distribution efforts $D_i$ to evaluate $a_i$ which is given by (see equation (40))

$$a_i = \frac{(p_i-c_i)a_id_i}{\sum_i (p_i-c_i)a_id_i}, \forall i$$

We now solve the posynomial program with equation (46) replaced by (47). With the resulting value of $D_i$ obtained, we reset $a_i$ from equation (48) and resolve the program with this new value of $a_i$. We repeat this procedure until there is no significant change in the solution.

5. Generalised Geometric Programming

Generalised geometric programming deals with the analysis of convex mathematical programs. Commonly these appear in the form:

Minimize $g_0(z)$

subject to $g_i(z) \leq 0$, $i \in I$ (50)

$z \in C$ (51)

where $C$ is a convex set and $g_i$, $i \in \{0\} \cup I$ are convex functions. In the generalised geometric programming approach, we first separate the arguments of the constraints and objective functions. We introduce the following notation:

$z = x^0 = x^i$, $\forall i \in I$

$x = x^0 \times x^i \in X$.

$X \triangleq \{x|x^0 = x^i, \forall i \in I\}$, a subspace of $X$

$C_i = C$, for $i \in \{0\} \cup I$

The program defined by equations (49), (50) and (51) may now be rewritten as:
Minimize $g_0(x^0)$
subject to $g_i(x^i) \leq 0 \quad i \in I$ (53)
$x^i \in C_i \quad i \in \{0\} \cup I$ (54)
$x \in X$ (55)

This formulation separates the constraints and the objective function excepting for the subspace condition, equation (55) which ties the problem together. The subspace captures the linearity of the problem.

At this stage we require further ideas relating to convexity. Recall from Section 1(iii), the definition of an epigraph to a pair $(g, C)$ of a convex function $g$ defined on a convex set $C$. Each nonvertical hyperplane that supports the epigraph of a convex function $g$ at a boundary point $(x', g(x'))$ produces a subgradient of $g$ at $x'$, i.e., a vector $\gamma \in \mathbb{R}^n$ with

$$g(x') + \langle \gamma, x - x' \rangle \leq g(x), \quad \forall x \in C$$ (56)

The subgradient set $\partial g(x')$ that consists of all such vectors $\gamma$ is generally a closed convex subset of $\mathbb{R}^n$ which contains only a single vector iff $g$ is differentiable at $x'$. In this case the single vector is the usual gradient vector $\nabla g(x')$.

A convex function $(g, C)$ is said to be closed if its epigraph is a closed set. We shall assume that all functions are closed.

The conjugate transform $(h, D)$ of an arbitrary convex function $(g, C)$ is defined by:

$$h(\gamma) = \sup_{x \in C} (\langle \gamma, x \rangle - g(x))$$ (57)
and

$$D = \{\gamma | \sup_{x \in C} (\langle \gamma, x \rangle - g(x)) < +\infty\}$$ (58)

We note that $h(\gamma) = \langle \gamma, x' \rangle - g(x')$ for each $\gamma \in \partial g(x')$.

By construction $(h, D)$ is a closed convex function defined on a convex set. A further consequence of the
conjugate transform is the conjugate inequality which states that

\[ g(x) + h(y) \geq \langle x, y \rangle \]  \hspace{1cm} (59)

for \( x \in C \) and \( y \in D \), with equality iff \( y \in \partial g(x) \) or equivalently \( x \in \partial h(y) \).

To handle constraints we require the conjugate transform of a particular function \( 0, g(x) \leq 0, x \in C \). This transform is related to the transform of \( [g, C] \) and is called the positive homogeneous extension of \( [h, D] \). This is denoted by \( [h^+, D^+] \) where

\[ h^+(y, \lambda) = \begin{cases} \lambda h(y/\lambda), & \lambda > 0 \\ \sup_{x \in C} \langle y, x \rangle, & \lambda = 0 \end{cases} \]  \hspace{1cm} (60)

and \( D^+ = \{(y, \lambda) | y/\lambda \in D, \lambda > 0\} \cup \{(y, 0) | \sup_{x \in C} \langle y, x \rangle < \infty\} \) \hspace{1cm} (61)

The conjugate inequality in this case is given by

\[ 0 + h^+(y, \lambda) \leq \langle x, y \rangle \]  \hspace{1cm} (62)

for \( (y, \lambda) \in D^+ \) and \( x \in C \) with equality iff \( y \in \lambda \partial g(x) \) or \( x \in \lambda \partial h^+(y, \lambda) \).

We now apply these ideas of conjugate transform theory to the program defined by equations (52) to (55). This program is the primal mathematical program. For the objective function, we have from equation (59) that

\[ g_0(x^0) + h_0(y^0) \geq \langle x^0, y^0 \rangle \]  \hspace{1cm} (63)

and for each constraint, from equation (62), we have that

\[ 0 + h_i^+(y_i^i, \lambda_i) \geq \langle x^i, y_i^i \rangle, \ i \in I \]  \hspace{1cm} (64)

Adding the inequalities (63) and (64), we have that

\[ g_0(x^0) + h_0(y^0) + \sum_{i \in I} h_i^+(y_i^i, \lambda_i) \geq \langle x, y \rangle \]  \hspace{1cm} (65)
where $x = x^0 \times x^i$ and $y = y^0 \times y^i$. By restricting $x \in X$ and $y \in X^-$, equation (65) becomes

$$g_0(x^0) + h_0(y^0) + \sum_{i \in I} h_i^+(y^i, \lambda_i) > 0 \quad (66)$$

with equality when

$$y^0 \in \partial g_0(x^0) \quad \text{or} \quad x^0 \in \partial h_0(y^0) \quad (67)$$

and

$$y^i \in \lambda_i \partial g_i(x^i) \quad \text{or} \quad x^i \in \partial h_i^+(y^i, \lambda_i), \ i \in I \quad (68)$$

Hence the geometric programming dual to the primal program (equations (52) to (55)) is given by:

$$\text{Minimize} \quad h_0(y^0) + \sum_{i \in I} h_i^+(y^i, \lambda_i) \quad (69)$$

subject to $y^0 \in D_0$, $(y^i, \lambda_i) \in D_i^+$, $i \in I \quad (70)$

$$y \in X^-$$

The dual program can have certain advantages over the original primal program:

(i) The nonlinear constraints are incorporated into the objective function.

(ii) If $X$ is of dimension $n$ and $X^-$ is of dimension $m$, then $X^-$, the orthogonal complement of $X$, is of dimension $n-m$. With the possibility of $n-m$ being much smaller than $m$ and the elimination of explicit non-linear constraints, the dual may turn out to be a much simpler problem to solve.

(iii) The fact that the primal and dual objectives sum to zero at optimality, i.e., $g_0(x^0) + h_0(y^0) + \sum_{i \in I} h_i^+(y^i, \lambda_i) = 0$, provides a good stopping criterion for an algorithm. The conditions for optimality are

$$x^0 \in C_0 \quad y^0 \in D_0$$
From these conditions one may calculate the optimal solution for one problem from the optimal solution of the other.

It is important to note the roles played by the four concepts of linearity, separability, convexity and duality in generalised geometric programming theory. Linearity may occur naturally in the problem, e.g., as linear constraints, or it may be induced by the need to separate the variables as in the program at the beginning of this section. Any linearity is usually conveniently captured in the subspace condition, equation (55). Separability is necessary to facilitate a simple computation of conjugate transforms. Convexity (and closure) guarantees that there is no duality gap, i.e., the primal and dual objectives sum to zero at optimality. Duality is the goal of generalised geometric programming theory.

Example

A well known problem in inventory control is to select a set \( \{x_t; \ t=1,\ldots,T\} \) of production levels to minimize, over a planning horizon of length \( T \), the sum of production and holding costs, whilst meeting demand. Formally the problem may be posed as:

\[
\text{Minimize} \quad \sum_{t=1}^{T} c(x_t) + h_t y_t
\]

subject to the inventory balance dynamics
\[ y_1 = x_1 - d_1 \]
\[ y_t - y_{t-1} = x_t - d_t, \quad t = 2, \ldots, T - 1 \]  
\[ -y_{T-1} = x_T - d_T \]

and the non-negativity constraints

\[ y_t \geq 0, \quad t = 1, \ldots, T \]
\[ x_t \geq 0, \quad t = 1, \ldots, T \]

Here \( y_t \) denotes the inventory level in period \( t \), \( d_t \) is the demand in period \( t \), \( c(x_t) \) is the production cost (assumed convex and strictly monotonically increasing) and \( h_t \) is the holding cost per unit in period \( t \).

To invoke the theory of generalised geometric programming, we need to put the constraint equations (73) into a subspace. Hence we introduce a new variable \( a_t \), \( t = 1, \ldots, T \), and restrict it to a one point domain \( \{ d_t \} \), \( t = 1, \ldots, T \). This variable is then associated with an additive component of the objective function which is identically zero. Hence we obtain a subspace condition

\[ y_1 - x_1 + a_1 = 0 \]
\[ y_t - y_{t-1} - x_t + a_t = 0, \quad t = 2, \ldots, T - 1 \]  
\[ -y_{T-1} - x_T + a_T = 0 \]  

(75)

It is convenient to treat the non-negativity constraints

\[ x_t \geq 0, \quad y_t \geq 0, \quad t = 1, \ldots, T, \]  

in an implicit way, i.e.,

\[ C_0 = \{ x_t \mid x_t \geq 0, \quad y_t \geq 0, \quad d_t = a_t, \quad t = 1, \ldots, T \} \]. Our problem is now in a form which is directly suitable for application of the theory. We note that in this problem there are no explicit constraints as in equation (53). The dual objective is given by
\[
\begin{align*}
\sup_{x_t, y_t, \alpha_t} \sum (x_t u_t + y_t v_t + \alpha_t \beta_t - c(x_t) - h_t y_t) \\
&= \sup_{x_t \geq 0, y_t \geq 0} \sum (x_t u_t - c(x_t)) + \sup \sum \alpha_t \beta_t + \sup \sum (y_t v_t - h_t y_t) \\
&= \sum c^*(u_t) + d_t \beta_t
\end{align*}
\]

where \(c^*(u_t) = \sup_{x_t \geq 0} x_t u_t - c(x_t)\)
\[u_t \in \mathcal{C}(x_t)\]
and \(v_t \leq h_t\)

Usually \(c(x_t)\) is a quadratic function and \(c^*(u_t)\) is readily calculated. Further we require the orthogonal complement to the subspace defined by equations (75), i.e.

\[\{(u_t, v_t, \beta_t) | \sum (u_t x_t + v_t y_t + \beta_t \alpha_t) = 0, \forall x_t, y_t, \alpha_t\}\]

satisfying equations (75). A straightforward calculation shows that

\[v_t = p_t - p_{t+1}, \quad t=1, \ldots, T-1\]
\[u_t = -p_t, \quad t=1, \ldots, T\]
\[\beta_t = p_t, \quad t=1, \ldots, T\]

Hence the dual problem is

\[
\begin{align*}
\text{Minimize} & \sum_{t=1}^{T} c^*(-p_t) + d_t p_t \\
\text{subject to} & \quad p_t - p_{t+1} - h_t \leq 0, \quad t=1, \ldots, T-1
\end{align*}
\]

For constant \(h_t\), we have a minimization over a monotonically increasing set of decision variables.
6. The Analysis of Posynomial Programming by Generalised Geometric Programming

Recall that in the preceding section we were able to use the convexity of the functions of a convex mathematical program by applying separability and then using linearity and duality to derive a potentially simpler problem. To bring this structure out in posynomial programming, we consider the following convex function:

\[ \sum_{i \in [k]} \delta_i \log \frac{\delta_i}{c_i} \]  
\[ \text{defined on } \delta_i > 0, \ i \in [k] \text{ and } \sum_{i \in [k]} \delta_i = 1 \]  

Here the parameters \( c_i > 0, \ i \in [k] \). The conjugate transform of the function defined by (76) taken over the set defined by equations (77) may be shown to be

\[ \log \sum_{i \in [k]} c_i \exp(z_i) \]  

Hence the conjugate inequality from equation (59) is:

\[ \sum_{i \in [k]} \delta_i \log \frac{\delta_i}{c_i} \geq \sum_{i \in [k]} \delta_i z_i \]  

with equality when

\[ \delta_i = \frac{c_i \exp(z_i)}{\sum_{i \in [k]} c_i \exp(z_i)} \]  

In order to handle constraints of the form

\[ \log \sum_{i \in [k]} c_i \exp(z_i) \leq 0 \]  

we require the positive homogeneous extension (see equation (60)) of function (76) defined over the set (77). This is given by

\[ \sum_{i \in [k]} \delta_i \log \frac{\delta_i}{(\lambda_k c_i)} \]
defined for \( \lambda_i = \sum_{\delta_i} \delta_i, \delta_i > 0, i \in [k] \) \hspace{1cm} (83)

To relate functions of the form (78) to posynomial geometric programs discussed in Section 3 we set

\[ x_j = \log t_j, \forall j \] \hspace{1cm} (84)

and

\[ z_i = \sum_{j=1}^{m} a_{ij} x_j, \forall i \] \hspace{1cm} (85)

in equations (14) and (15). Equation (85) is the subspace condition in the generalised theory (see equation (55)). Making the above substitution and taking logs, our posynomial program is as follows:

\[
\begin{align*}
\text{Minimize} & \quad \sum_{i \in [0]} \log \sum_{i \in [k]} c_i \exp(z_i) \\
\text{subject to} & \quad \log \sum_{i \in [k]} c_i \exp(z_i) \leq 0, i=1,\ldots,p \\
& \quad z \in \chi = \{z_i \mid z_i = \sum_{j=1}^{m} a_{ij} x_j, i=1,\ldots,n\}
\end{align*}
\]

In the above form, the theory of generalised geometric programming may be invoked since we have already calculated the relevant conjugate transforms (see equations (76), (77), (82) and (83)). Hence, using equations (69), (70) and (71), the dual to a posynomial program is given by

\[
\begin{align*}
\text{Minimize} & \quad \sum_{i=1}^{p} \delta_i \log \delta_i/c_i - \sum_{k=1}^{p} \lambda_k \log \lambda_k \\
\text{subject to} & \quad \sum_{i \in [0]} \delta_i = 1 \\
& \quad \delta_i > 0, \forall i \\
& \quad \lambda_k = \sum_{i \in [k]} \delta_i, k=1,\ldots,p \\
& \quad \delta \in \chi^* = \{\delta_n \mid \sum_{j=1}^{n} a_{ij} \delta_i = 0, j=1,\ldots,m\}
\end{align*}
\]

The above objective function is equivalent to maximizing
which is the original form of the dual objective function for posynomial programming (see equations (25) to (28)). Further it is straightforward to obtain the optimality conditions between the primal and dual variables, equation (29) from the general optimality conditions in Section 5.

7. Applications

In the previous sections we have shown the development of the theory of geometric programming. Included in the references are various books and papers describing applications of geometric programming. In the sequel to this paper (appearing in the next issue of NZOR) we present in some detail the major areas of application of geometric programming.

REFERENCES


