ELSEVIER

European Journal of Operational Research 103 (1997) 230–241

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

Theory and Methodology

Investigation of path-following algorithms for signomial geometric programming problems

Hsu-Hao Yang [a,*], Dennis L. Bricker [b]

[a] *Department of Industrial Engineering and Management, National Chin-Yi Institute of Technology, Taiping, Taichung, Taiwan 41111*
[b] *Department of Industrial Engineering, University of Iowa, Iowa City, IA 52242, USA*

Received 26 June 1995; accepted 23 March 1996

Abstract

This paper considers signomial geometric programming (GP) dual problems, a class of nonconvex nonlinear programming problems possessing multiple locally optimal solutions. The primary purpose of this paper is to investigate the quality of solutions found by use of a path-following algorithm. The path-following method may be applied to either the original nonconvex problem, or to each of a sequence of convex posynomial GP problems approximating the original problem. For each test problem, the algorithms were initiated with thousands of different starting points. It was determined that, when the stopping criterion was relaxed for early posynomial GP problems in the sequence, the ultimate solution tended to be of better quality, and more frequently globally optimal. © 1997 Elsevier Science B.V.

Keywords: Interior point methods; Path-following algorithm; Geometric programming

1. Introduction

Geometric programming (GP) [5] is a class of optimization problems with a nonlinear objective function subject to nonlinear constraints. The term "geometric" is used because the arithmetic-geometric mean inequality plays a central role in its early development. One of the remarkable properties of GP is that if the primal problem is in posynomial form, then a global minimizing solution to that problem can be obtained by solving the dual maximization problem. Hence, any solution to the dual problem satisfying the necessary conditions corresponds to a global minimizing solution to the primal problem. Unfortunately, a signomial GP problem, which is a class of nonconvex nonlinear programming problems possessing multiple locally optimal solutions, does not have the inherent primal convexity, and hence greatly weakens the relationship between the primal and the dual problems. Nonetheless, it is still possible to formulate a dual problem corresponding to the primal problem. The drawback is that precise equivalence between primal and dual problems no longer holds, so that the dual problem is often referred to as "pseudo-dual".

The path-following algorithm (PFA) is one of the interior point methods which is inspired by Karmarkar's method [11]. If one is to minimize the objective function, the main idea behind the interior point methods is that the objective function should

* Corresponding author. Email: yanghh@chinyi.ncit.edu.tw.

PII S0377-2217(96)00265-2

be minimized and the solutions kept away from the boundary. The classical way to implement this idea is to use a barrier function to solve a sequence of subproblems associated with a positive barrier parameter. As the barrier parameter gradually decreases by a certain factor and approaches zero, the sequence of the solutions will converge to an optimal solution.

In this paper, we investigate the use of the path-following algorithm on signomial GP problems. The main purpose of the research emphasizes more on the quality of the solution than the time required to find the solution; therefore, the CPU time will not be a major issue of concern. The computational results are based on the implementation of the algorithms using APL (A Programming Language) code, either on a Macintosh SE or an HP-UX workstation.

This paper is organized as follows. Section 2 introduces the primal-dual pair of GP problems and reviews the literature. Section 3 describes how to apply the path-following algorithm to solving the posynomial GP dual problem. Section 4 is the motivation of applying the path-following algorithm in this research. Computational results of different strategies for solving signomial GP problems are presented in Section 5. Section 6 contains the conclusions.

2. The primal-dual pair of GP

A GP primal problem can be expressed as the following general form

Minimize $g_0(t)$

subject to

$g_k(t) \le 1, \quad k = 1,2,\ldots,p,$

$t_j > 0, \quad j = 1,2,\ldots,m,$

where

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

$$I = (1,2,\ldots,n), \quad [k] \subseteq I, \quad \bigcup_{k\ 0}^{p} [k] = I,$$

and $[k] \cap [s] = \varnothing$ for $k \neq s$,

$\sigma_i = \pm 1$ for all $i \in I$,

$c_i > 0$ for all $i \in I$,

a_{ij} unrestricted in sign for all $i \in I$ and $j = 1,\ldots,m$.

The index set I numbers the total n terms, and the index subset $[k]$ numbers the terms in a function g_k. In each g_k, if σ_i's are all positive, then this g_k is referred to as a posynomial. On the other hand, if one or more of σ_i's are negative, then function g_k is called a signomial. A GP problem is a posynomial GP problem if each g_k it contains is a posynomial, otherwise it is a signomial GP problem.

The posynomial GP primal problem can be simplified as follows

Minimize $g_0(t)$

subject to

$g_k(t) \le 1, \quad k = 1,2,\ldots,p,$

$t_j > 0, \quad j = 1,2,\ldots,m,$

where

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

Corresponding to a posynomial GP primal problem, the GP dual problem is to

$$\text{Maximize } v(\delta,\lambda) = \prod_{k=0}^{p} \left\{ \lambda_k^{\lambda_k} \prod_{i\in[k]} \left(\frac{c_i}{\delta_i} \right)^{\delta_i} \right\}$$

subject to

$\lambda_0 = 1,$

$$\sum_{i\in I} a_{ij}\delta_i = 0, \quad j = 1,2,\ldots,m,$$

$$\sum_{i\subset[k]} \delta_i = \lambda_k, \quad k = 0,1,\ldots,p,$$

$\delta_i \ge 0$ for all $i \in I$,

$\lambda_k \ge 0,$

which, unlike the primal, is linearly constrained.

The λ variable is not really a decision variable since each λ_k is simply the sum of the δ_i values for all the terms in that posynomial k and may be eliminated by substitution. The logarithm of the objective function $v(\delta,\lambda)$ is a concave function if λ is eliminated as suggested. Since the logarithmic func-

tion is strictly increasing, the maxima of $v(\delta,\lambda)$ and its logarithmic function will be achieved at the same dual solution. One thus could maximize the logarithm of the dual objective function

$$\text{Maximize } f(\delta) = \ln(v(\delta)) = \sum_{i=1}^{n} \{\delta_i \ln c_i - \delta_i \ln \delta_i\} + \sum_{k=0}^{p} \left(\sum_{i\in[k]} \delta_i\right) \ln\left(\sum_{i\in[k]} \delta_i\right)$$

subject to

$$\lambda_0 = 1,$$

$$\sum_{i\in I} a_{ij}\delta_i = 0, \quad j = 1,2,\ldots,m,$$

$$\delta_i \geq 0 \quad \text{for all } i \in I.$$

The relationship between optimal dual and primal solutions, namely

$$\delta_i g_k(t) = \lambda_k c_i \prod_{j=1}^{m} t_j^{a_{ij}}, \; i \in [k], \; k = 0,1,\ldots,p,$$

can in most cases be used to recover the primal optimal solution when a dual-based algorithm is used. If a primal constraint k is slack at the optimum, all the optimal dual variables λ_k and δ_i ($i \in [k]$) associated with this constraint are zero. Dual-based algorithms typically include procedures for special handling of slack primal constraints, e.g., placing a lower positive bound on all dual variables.

The pseudo-dual signomial GP problem is to

$$\text{Maximize } v(\delta,\lambda) = \sigma_0 \prod_{k=0}^{p} \left\{ \lambda_k^{\lambda_k} \prod_{i\in[k]} \left(\frac{c_i}{\delta_i}\right)^{\sigma_i\delta_i} \right\}^{\sigma_0}$$

subject to

$$\sum_{i\in[0]} \sigma_i\delta_i = \sigma_0,$$

$$\sum_{i\in I} \sigma_i a_{ij}\delta_i = 0, \quad j = 1,2,\ldots,m,$$

$$\sum_{i\in[k]} \sigma_i\delta_i = \lambda_k, \quad k = 0,1,\ldots,p,$$

$$\delta_i \geq 0 \quad \text{for all } i \in I,$$

$$\lambda_k \geq 0.$$

The value of σ_0 is the sign of the optimal value, which will usually be known in advance for most problems. Linear constraints still constitute the constraint set; however, the logarithm of the objective function is not a concave function over the feasible region. In fact, the weak duality theorem for the posynomial GP problem does not hold true for this dual, and hence it is called a pseudo-dual.

2.1. Literature review for signomial GP problems

A GP problem can be solved in its either primal or dual form. Among all primal-based algorithms for signomial GP problems, successive approximation by posynomials, called "condensation", has received the most popularity. The idea is to approximate a signomial GP problem at the current iteration by condensing to an approximating posynomial problem. Thus the solution of the signomial GP problem proceeds by iteratively solving a sequence of posynomial approximations, for which the solutions converge to at least a stationary point of the signomial problems. The main difference between the various algorithms ([1,4,10,17,19,23]) which employed this solution philosophy lies in the precise forms of the posynomial approximations to signomial programs.

Since a signomial GP problem has no convex reformulation, Passy and Wilde [18] developed a weaker type of duality, called pseudo-duality, to accommodate this class of nonlinear optimization. The pseudo-duality theorem assures the occurrence of a stationary point, called "pseudo-maximum", in the pseudo-dual program corresponding to every local minimum of the primal problem such that the primal and pseudo-dual objective values are equal. To find the global solution one must find all locally maximizing solutions and then choose, not the maximum, but the minimum, from among them. This procedure is called "pseudo-minimization". Other dual-based algorithms were developed in [2] and [21].

Use of general purpose nonlinear programming methods taking no advantage of the structure of GP problems were reported in [7,8,22,25]. Discussions on computational aspects of solving signomial GP problems are available in [3] and [24]. A detailed survey of several techniques for solving signomial programs can be found in [20]. The comparison of a special purpose algorithm versus general purpose

algorithms was published by Ecker et al. [6]. Their experiments showed that the special purpose algorithm often finds approximate solutions more quickly than do the general purpose algorithms, but is usually not significantly more efficient when greater accuracy is required.

2.2. Condensation of signomial GP problems

Since the condensation process is the most popular method to solve the signomial GP problem, this section is dedicated to showing how a signomial GP problem can be approximated by posynomial programs. The principle is to approximate a multiterm posynomial with a monomial, i.e., a single term.

Consider again the general form of the GP primal problem

Minimize $g_0(t)$

subject to

$$g_k(t) \le 1, \quad k = 1,2,\ldots,p,$$

$$t_j > 0, \quad j = 1,2,\ldots,m.$$

Assume that $g_0(t) > 0$ at the optimum. By introducing a new variable t_0 and adding the constraint $t_0 \ge g_0(t)$, it can be rewritten as the following general form

Minimize t_0

subject to

$$g'_k(t) \le 1, \quad k = 0,1,\ldots,p,$$

where

$$g'_k(t) = \sum_{i\in[k]} \sigma_i c_i \prod_{j=1}^{m} t_j^{a_{ij}}, \quad k = 1,2,\ldots,p,$$

$$g'_0(t) = \sum_{i\in[0]} \sigma_i c_i \prod_{j=0}^{m} t_j^{a_{ij}},$$

where

$$a_{i0} = -1.$$

Furthermore, $g'_k(t)$ can be written as the difference of two posynomials

$$g'_k(t) = P_k(t) - Q_k(t),$$

where

$$P_k(t) = \sum_{i\in[k]^+} u_i = \text{collection of positive terms in the } k\text{-th constraint},$$

$$Q_k(t) = \sum_{i\in[k]^-} u_i = \text{collection of negative terms in the } k\text{-th constraint},$$

$$[k]^+ = \{i \mid \sigma_i = +1 \text{ for } i \in [k]\}, \quad [k]^- = \{i \mid \sigma_i = -1 \text{ for } i \in [k]\},$$

$$u_i = \begin{cases} c_i \prod_{j=0}^{m} t_j^{a_{ij}}, & i \in \begin{cases} [0]^+ & \text{if } \sigma_i = +1 \\ [0]^- & \text{if } \sigma_i = -1 \end{cases} \\ c_i \prod_{j=1}^{m} t_j^{a_{ij}}, & i \in \begin{cases} [k]^+ & \text{if } \sigma_i = +1 \\ [k]^- & \text{if } \sigma_i = -1 \end{cases} \end{cases}.$$

Therefore, one can derive the following equivalent program

Minimize t_0

subject to

$$\frac{P_k(t)}{Q_k^+(t)} \le 1, \quad k = 0,1,\ldots,p,$$

where

$$Q_k^+(t) = 1 + Q_k(t).$$

Use of the arithmetic-geometric mean inequality with $Q_k^+(t) = \sum_i u_i$ results in the following

$$Q_k^+(t) \ge \prod_i \left(\frac{u_i(t)}{\delta_i} \right)^{\delta_i}, \quad i \in [k],$$

where

$$\delta_i = \frac{u_i(t')}{Q_k^-(t')}, \ i \in [k], \ t' \text{ is a current operating point.}$$

If $P_k(t)$ is divided by both sides, then it implies that

$$\frac{P_k(t)}{Q_k^+(t)} \le \frac{P_k(t)}{\prod_i \left(\frac{u_i(t)}{\delta_i} \right)^{\delta_i}}, \quad i \in [k],$$

where the function on the right hand side is a posynomial. Let $Q'_k(t)$ represent the monomial obtained by condensing $Q_k^+(t)$ at the point t', one has the following problem

Minimize t_0

subject to

$$\frac{P_k(t)}{Q'_k(t)} \le 1, \quad k = 0,1,\ldots,p.$$

Note that posynomial constraint

$$\frac{P_k(t)}{Q'_k(t)} = \frac{P_k(t)}{\prod_i \left(\frac{u_i(t)}{\delta_i}\right)^{\delta_i}} \le 1$$

implies that

$$\frac{P_k(t)}{Q_k^+(t)} \le 1,$$

since $Q_k^+(t) \ge Q'_k(t)$. That is, the feasible region of the condensed posynomial constraints is a subset of the feasible region of the original signomial constraints. Hence, any optimal solution, t, to the condensed program will be feasible (but not necessarily optimal) for the original problem. The condensed posynomial and that original posynomial are equal at the point $t = t'$.

3. Path-following algorithms

Consider the following linearly constrained convex programming problem with non-negative variables

Minimize $f(x)$

subject to

$Ax = b$,

$x \ge 0$,

where

$f(x)$ is a twice differentiable function,

A is an $m \times n$ matrix,

$x = (x_1,\ldots,x_n)^{\mathrm{T}}$ is an n-vector,

$b = (b_1,\ldots,b_m)^{\mathrm{T}}$ is an m-vector.

Assume that the interior feasible set $S \equiv \{x \in R^n;\ Ax = b,\ x > 0\}$ is nonempty and bounded. If a logarithmic term is added, then one has to solve the following family of problems

$$\phi(z) = \text{Minimize } f(x) - z\sum_i \ln x_i$$

subject to

$Ax = b$,

where

$z > 0$ is the barrier penalty parameter,

$\sum_i \ln x_i$ serves as a barrier function.

As one solves the problem penalized by the barrier function for a sequence of decreasing positive values of the parameter z, the result is a sequence of feasible points converging to the optimal solution of the original problem.

The Karush–Kuhn–Tucker (KKT) optimality conditions for $\phi(z)$, which must be satisfied by $x(z)$, the optimal solution of $\phi(z)$, are written as:

$$\begin{cases} Xs = ze \\ Ax = b \end{cases},$$

where

X is the $n \times n$ diagonal matrix of x,

$s = \nabla f(x) - A^{\mathrm{T}}y$,

$\nabla f(x)$ is the gradient of $f(x)$,

y are dual variables associated with the constraints

$Ax = b$,

$e = (1,1,\ldots,1)^{\mathrm{T}}$.

Let Γ denote $\{(x,s):\ Xs = ze,\ z > 0\}$. The trajectory Γ, for $z > 0$, is known as the central path because of the interiority forced by the barrier function. The path-following algorithm generates a sequence of solutions, (x^k,s^k), which lie in the proximity of the central trajectory of the problem.

One can use Newton's method to solve the KKT optimality conditions, which is a system of nonlinear equations. If Newton's method is applied, suppose that at iteration k, one has the solution x^k, y^k, s^k;

then one must solve the following linear system, resulting in search direction $(\Delta x, \Delta y)$

$$\begin{cases} \left[\nabla^2 f(x^k) + (X^k)^{-1} S^k\right] \Delta x - A^T \Delta y \\ \quad = \gamma (X^k)^{-1} e - s^k \\ A \Delta x = 0 \end{cases},$$

where

$(X^k)^{-1}$ is the $n \times n$ diagonal matrix

with elements $\dfrac{1}{x_i^k}$,

S^k is the $n \times n$ diagonal matrix with elements s^k,

$\nabla^2 f(x)$ is the Hessian matrix of $f(x)$,

$\gamma = (1 - \alpha) z^k < z^k$,

$0 < \alpha < 1$.

Once the search direction $(\Delta x, \Delta y)$ is computed, let

$$x^{k+1} = x^k + \Delta x,$$

$$y^{k+1} = y^k + \Delta y,$$

$$s^{k+1} = \nabla f(x^{k+1}) - A^T y^{k+1},$$

thus completing one iteration of the algorithm. Rather than making several Newton's steps to converge to the central path for a fixed value of z, one iteration only is usually performed of Newton's method, and then z is reduced.

In the previous discussion, it is assumed that an initial feasible solution is available so that one can start the algorithm. However, it is necessary in general to produce such an solution so that the algorithm can be initiated. An initial feasible solution can be obtained by the introduction of an artificial variable and by rewriting the objective function and constraints as follows

$$\text{Minimize } f(x) - z \sum_i \ln x_i + M x_a$$

subject to

$$Ax + (b - Ax^0) x_a = b,$$

$$\sum_{i \neq a} x_i \leq M,$$

where

M is a positive big cost,

x_a is an artificial primal variable,

$x^0 > 0$ is an initial solution,

$\sum_{i \neq a} x_i \leq M$ is the bounding row which will correspond to the artificial dual variable.

Now (x^0, x_a) is feasible with $x_a = 1$. In our implementation, we let x^0 equal to $e = (1,1,\ldots,1)$. Once an initial feasible solution (x^0, x_a) is available, x_a must be driven to zero at the optimum solution. In order to obtain values of the y variables for the initial solution, we let each of them equal to zero except the artificial variable which is associated with the bounding row.

3.1. Applying PFA to GP dual problems

The path-following algorithm applied to a twice differentiable convex function $f(x)$ solves a sequence of problems of the form

$$\phi(z) = \text{Minimize } f(x) - z \sum_i \ln x_i$$

subject to

$$Ax = b,\ x \geq 0.$$

To apply the path-following algorithm, we need to minimize the negative of $\ln(v(\delta))$ so that it is in standard form

$$\phi(z) = \text{Minimize } -f(\delta) - z \sum_i \ln \delta_i$$

subject to

$$A\delta = b,$$

where

$$A = \begin{bmatrix} 1\,1\ldots1\,0\,0\ldots0 \\ A^T \end{bmatrix}, \quad A^T\delta = \sum_{i \in I} a_{ij} \delta_i = 0,$$

$j = 0,1,\ldots,m$,

$\delta = (\delta_1,\ldots,\delta_n)^T$, $\delta_i \geq 0$ for all i,

$b = (\lambda_0, 0,\ldots,0)^T$, $\lambda_0 = 1$.

The system of equations of optimality conditions needed to be solved for a fixed z is of the form

$$\begin{cases} Ds = ze \\ A\delta = b \end{cases},$$

where D is the diagonal matrix of δ,

$$s = \nabla(-f(\delta)) - A^{T}y.$$

Other than linear programming (LP), extensions of interior point methods to special classes of programs are available in [9,12–16].

4. Motivation of applying path-following algorithm

The major motivation of using the path-following algorithm for solving a problem possessing many local solutions is that the algorithm is initiated with an interior point and moves within the feasible region. Intuitively, if the solution found by the path-following algorithm for each condensed posynomial program is not too close to the boundary, then it should have a better opportunity to avoid being trapped in competing local solutions in the early iterations of the algorithm. What this implies is that it is hoped that the algorithm can approach the global solution eventually, but not too early. The question lies in how to implement the path-following algorithm so that the solution will not converge to one of the competing local minima in the early iterations. The answer is at hand in the termination tolerance of the path-following algorithm. Remember that the path-following algorithm is terminated when the value of z, which is the barrier parameter, is less than the tolerance. If the tolerance is not small enough, the algorithm terminates at a point which may not really satisfy the optimality conditions, while could be a desirable outcome for our purpose, in that the solution is far away from the boundary. Of course, the tolerance needs to be small enough at the final solution so that the algorithm converges to a point which satisfies the optimality condition. The convergence proof of this approach is given in [26].

How could the initial value of the termination tolerance, which is reduced during the course of the optimization, affect the quality of the solution? For example, if the algorithm is initiated with the same starting point but with two distinct tolerances, will the algorithm converge to different local solutions? What can be characterized under several tolerances using the same starting point? For that reason, the implemented algorithm uses the initial tolerance (*z_tol*) as an input. After the tolerance is input, the problem is condensed to an approximating posynomial which is then solved by the path-following algorithm. As each condensed posynomial is solved, the termination tolerance of the path-following algorithm is reduced by a certain factor (*z_factor*), which is positive and less than 1. By the reduction of this factor, the tolerance would gradually decrease, converging to zero. The algorithm iteratively proceeds as follows.

Step 0. Rewrite the problem as a linear objective function constrained by upper bounds of 1 on nonlinear functions expressed as a ratio of posynomial functions. Select an initial point t' (which need not be feasible) and termination tolerance, *z_tol*, of the path-following algorithm.

Step 1. Condense the denominator of each ratio, which is a posynomial function, into a single term, using weights evaluated at the current point t'.

Step 2. Solve the resulting condensed posynomial program by applying the path-following algo-

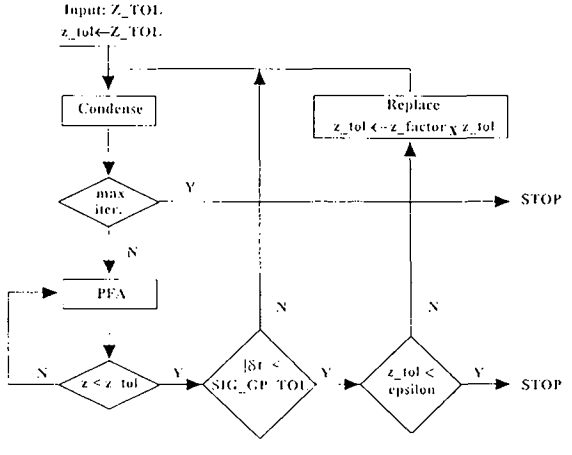

Fig. 1. Algorithm for solving a sequence of condensed posynomials by PFA.

Table 1
Number of global solutions found using two different initial *z_tols* of PFA

Problem no.	Total no. of variables	Total no. of constraints	Total no. of terms in condensed GP	*z_tol* = 0.01		*z_tol* = 0.0001	
				freq.	%	freq.	%
*t*1	2	4	6	1483	44.08	898	26.97
*t*2	2	4	6	2273	67.57	1143	33.98
*t*3	2	4	5	2277	67.59	2295	68.22
*t*4	2	4	6	2322	69.02	2295	68.22
*t*5	2	4	6	1070	31.81	1551	46.11

rithm with termination tolerance *z_tol*. Calculate the new solution t''.

Step 3a. Compare the distance moved, δt, to some tolerance. If the tolerance is satisfied, go to Step 3b; otherwise, let $t' = t''$ and go to Step 1.

Step 3b. If *z_tol* is less than the stopping criterion, epsilon, stop; otherwise, go to Step 4.

Step 4. Reduce *z_tol* by a certain factor, let $t' = t'$ and go to Step 1.

Fig. 1 shows the proposed algorithm. The distance moved in Step 3a is measured by $\delta t = \sum_{j=1}^{m} |t_j'' - t_j'|$. The stopping criterion, epsilon, in Step 3b, is limited by the precision of the computations; a value of 10^{-14} was used in obtaining the computational results which follow.

5. Results of computational strategies for signomial GP problems

In this section, the following situations will be investigated

1. solve a sequence of condensed posynomial dual problems by PFA,
2. solve the pseudo-dual signomial problems by PFA.

5.1. Solve a sequence of condensed posynomial dual problems by PFA

To investigate the effect of the accuracy with which the earlier approximating posynomial GP problems are solved, five test problems (Table 1) were artificially generated, each having several competing locally optimal solutions. (Appendix A in [26] lists these problems and their solutions.) For each problem, a total of 3364 starting points are selected. These 3364 points are uniformly distributed with increment equal to 0.4 in a square region.

For each starting point, the problem was solved twice, once with the initial stopping tolerance (*z_tol*) equal to 0.01 and again with the tolerance equal to 0.0001. In each case, the tolerance was reduced by a factor of 0.01, with the algorithm terminated after an approximating posynomial GP problem was solved with *z_tol* equal to 10^{-14}.

To have a better picture of how the algorithm proceeds, test problem *t*1 is used as an illustration.

Minimize $g_0 = t_2$

subject to

$$g_1 = -0.0019t_1^2 + 0.09108t_1 - 0.0038t_2 \le 1,$$

$$g_2 = -0.0013t_1^2 + 0.0779t_1 - 0.0065t_2 \le 1,$$

$$g_3 = -0.000814t_1^2 + 0.0586t_1 - 0.002035t_2 \le 1,$$

$$g_4 = 0.1t_1^2t_2^{-1} - 5.8t_1t_2^{-1} + 105.1t_2^{-1} \le 1.$$

The feasible region of the test problem *t*1 is shown in Fig. 2, where several local solutions are present.

Now, let's take a closer look at the step at which the first condensed problem was just solved by the path-following algorithm, starting from a feasible point (37, 37). In Fig. 3, it appears that with a smaller initial *z_tol*, the algorithm has the tendency to go to a local solution in the early stage of the algorithm. This can be further verified if the interested point was forwarded to the solution where the second condensed problem was solved, as shown in Fig. 4.

By studying Fig. 3 and Fig. 4, one can find significant clues that suggest that use of a larger tolerance at the beginning could be advantageous. To

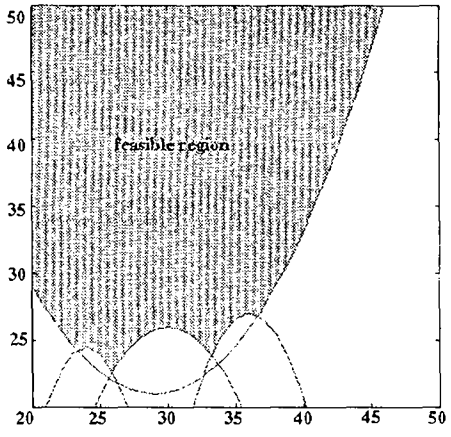

Fig. 2. Feasible region of the test problem $t1$.

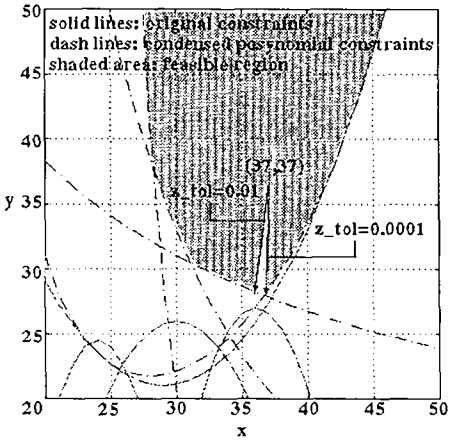

Fig. 3. Path-following where the first condensed program is solved starting from a point with two different initial *z_tols*.

identify the dependence of the local solution found on the locations of all starting points, the basins of attraction for two *z_tols* are shown in [26]. The basin of attraction of a local solution is defined as the set of different starting points from which the algorithm converges to that solution. When a larger initial *z_tol* (0.01) is used, there are only two basins of attraction, including that of the global solution. If a smaller initial *z_tol* (0.0001) is used, however, basins of attractions appear for other competing local solutions. This encouraging result justifies our use of the termination tolerance.

Having justified our research purpose, we further compare our algorithm with two existing methods which are commonly referred in the literature. Table 2 lists the test problems and their solutions, where problem names beginning with "RM" are from [24], "D" from [3]; "n/a" means not available; total

Table 2
The results by PFA compared to existing methods

Problem name	V	C	Primal objective value			Total infeasibilities of constraints		
			PFA	RM	D	PFA	RM	D
RM09	2	1	11.96438	11.96392	n/a	0	0.000004965	n/a
RM10	3	1	−83.24973	−83.05711	n/a	0	0	n/a
RM11	4	2	−5.73982	−5.73972	n/a	0	0	n/a
RM12	8	4	−6.04823	−6.04823	n/a	0	0.000001384	n/a
RM13(D5)	8	6	7049.2477	7049.247	7049.324	0	0.000004616	0.000006212
RM14	10	7	1.14362	1.14368	n/a	0	0	n/a
RM15	10	7	0.20565	0.2056	n/a	0	0.00136019	n/a
RM16	10	7	0.19663	0.19659	n/a	0	0.00179802	n/a
RM17	11	9	0.14061	0.14061	n/a	0	0.009473293	n/a
RM18	13	9	1.86163	1.81818	n/a	0	2.374294926	n/a
RM19	8	5	17485.988	17486.039	n/a	0	0.000001522	n/a
RM20	13	9	−116.4515	−121.5388	n/a	0	0.03119618	n/a
RM22	9	10	376.3109	−375.7705	n/a	0	0.001218221	n/a
RM23(D2)	5	6	10121.773	10121.977	10122.431	0	0.000062905	0
RM24(D6)	10	10	97.59237	97.59753	97.59103	0	0.000186638	0.000019081
D4A	8	4	3.95116	n/a	3.9517	0	n/a	0.000016337

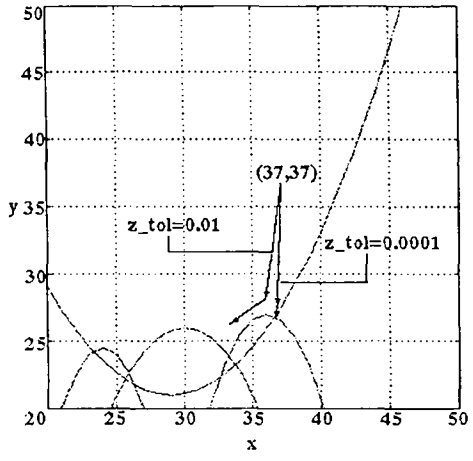

Fig. 4. Path-following where the second condensed program is solved starting from a point with two different initial *z_tols*.

infeasibilities of constraints are computed by $\Sigma_{k \in K}(g_k - 1)$, $K = \{k \mid g_k > 1, \quad k = 1,2,\ldots,p\}$; "$V$" is the number of primal variables, and "C" is the number of primal constraints. According to Table 2, it can be seen that the PFA outperforms two existing methods in terms of primal objective value and total infeasibilities of constraints. Unfortunately, however, only a single instance in the literature was found for which a signomial GP problem has several competing local solutions, namely test problem D4A in [3]. This problem was initiated with 2304 primal starting points. It was found that when a larger initial tolerance ($z\_tol = 10^{-4}$) was used, 2296 points (99.65%) converge to the global solution, while only 2284 points (99.13%) converge to the global solution when using a smaller initial tolerance ($z\_tol = 10^{-8}$).

5.2. Solve pseudo-dual signomial problems by PFA

The results (Table 4.3 in [26]) suggest that the approach is not recommended. The main drawbacks are

1. for most of the test problems, the algorithm did not converge,
2. even when the algorithm converges, the solution is not guaranteed to be a global solution,
3. the number of iterations required to converge is large, making it less competitive.

The major problem associated with this approach is that the solution generated by the path-following algorithm frequently tends to approach the boundary or even to leave the feasible region. This tendency not only makes it difficult for the algorithm to approach a feasible solution at each iteration, but slows down convergence because the actual step size needs to be shortened to maintain feasibility. Although several stepsize reduction factors, ranging from 0.1 to 0.9, were tested in an attempt not to approach too close to the boundary, none worked well. Another alternative is to use a two-phase method to find a feasible solution for the path-following algorithm. The intention was, rather than always initiating the search with the unit vector, to start the algorithm from a solution which is feasible and likely to be located inside the feasible region. However, the results show this alternative to be not helpful.

6. Conclusions

This research primarily investigates the use of a path-following algorithm for solving signomial GP problems, whose special feature is possessing multiple locally optimal solutions. Two approaches for solving the problem were investigated, i.e., either solving a sequence of approximating posynomial convex programs or solving its pseudo-dual directly. The research leads to the following major conclusions:

1. Solving the signomial GP problems by iteratively solving the dual problems of a sequence of approximating posynomial GP problems is apparently better than solving its pseudo-dual directly.
2. When applying the path-following algorithm to a problem with many locally optimal solutions, in order to converge to a better quality solution, the algorithm should not terminate at a point that is too close to the boundary of the first few approximating posynomial GP problems.

There exists limitations regarding the application of the proposed method, too. First, the results are solely derived from dealing with signomial GP problems. In addition, one might ask, how "close" is too

close? At this moment, no rule can be explicitly given, however. Above all, both the size and the structure of the problem play a crucial role in determining the termination tolerance.

Second, from the perspective of computational efficiency, when Newton's method was used to solve the following system of nonlinear equations

$$\begin{cases} Xs = ze \\ Ax = b \end{cases},$$

it requires the inverse of this approximating linear system. Occasionally, it is difficult to compute the inverse, especially when the solutions are highly sensitive to the capability of a computer. In our experience, the same APL codes implemented on a Macintosh SE and an HP-UX workstation could return different results when the system equations are near-singular. To deal with this difficulty, techniques such as the conjugate gradient method were utilized when the inverse is not obtainable. One can see that the implementation of this technique is at the cost of computational efforts.

Finally, instead of specifying the termination tolerance of the path-following algorithm, another alternative is to specify the maximum number of iterations performed at each iteration of the path-following algorithm. We favor using the tolerance rather than maximum iterations as a termination criterion, since the value of tolerance, z, is the barrier parameter and would serve as a better indicator of how close the point is to the boundary.

Acknowledgements

The first author was supported in part by National Science Foundation grant no. 85-2213-E-158-002.

References

[1] Avriel, M., and Williams, A.C., "Complementary geometric programming", *SIAM Journal on Applied Mathematics* 19 (1970) 125–141.

[2] Blau, G.E., and Wilde, D.J., "A Lagrangian algorithm for equality constrained generalized polynomial optimization", *A.I.Ch.E. Journal* 17 (1971) 235–240.

[3] Dembo, R.S., "Current state of the art of algorithms and computer software for geometric programming", *Journal of Optimization Theory and Applications* 26/2 (1978) 149–184.

[4] Duffin, R.J., and Peterson, E.L., "Reversed geometric programming treated by harmonic means", *Indiana University Mathematics Journal* 22 (1972) 531–550.

[5] Duffin, R.J., Peterson, E.L., and Zener, C.M., *Geometric Programming*, John Wiley, New York, 1967.

[6] Ecker, J.G., Kupeerschmid, M., and Sacher, R.S., "Comparison of a special-purpose algorithm with general-purpose algorithms for solving geometric programming problems", *Journal of Optimization Theory and Applications* 43/2 (1984) 237–263.

[7] Falk, J.E., "Global solutions of signomial problems", Report No. T-274, George Washington University, 1973.

[8] Fiacco, A.V., and Ghaemi, A., "Optimal treatment levels of a stream pollution abatement system under three environmental control policies", Report No. T-38, George Washington University, 1979.

[9] Han, C., Pardalos, P.M., and Ye, Y., "Implementation of interior-point algorithms for some entropy optimization problems", *Optimization Methods and Software* 1/1 (1992) 71–80.

[10] Jefferson, T., "Manual for the geometric programming code GPROG (CDC) version 2", Report No. 1974/OR/2, Mechanical and Industrial Engineering Department, University of New South Wales, Australia, 1974.

[11] Karmarkar, N., "A new polynomial-time algorithm for linear programming", *Combinatorica* 4 (1984) 373–395.

[12] Karmarkar, N., Resende, M.G.C., and Ramakrishnan, K.G., "Interior point algorithm to solve computationally difficult set covering problems", *Mathematical Programming* 52/3 (1991) 597–618.

[13] Kojima, M., Megiddo, N., and Ye, Y., "An interior point potential reduction algorithm for the linear complementarity problem", *Mathematical Programming* 54 (1992) 267–279.

[14] Kortanek, K.O., and No, H., "A second order affine scaling algorithm for the geometric programming dual with logarithmic barrier", *Optimization* 23 (1992) 303–322.

[15] Kortanek, K.O., and Zhu, J., "A polynomial barrier algorithm for linearly constrained convex programming problems", *Mathematics of Operations Research* 18/1 (1993) 116–127.

[16] Monteiro, R.C., and Adler, I., "Interior path following primal-dual algorithms. Part II: convex quadratic programming", *Mathematical Programming* 44 (1989) 43–66.

[17] Pascual, L., and Ben-Israel, A., "Constrained maximization of posynomials by geometric programming", *Journal of Optimization Theory and Applications* 5 (1970) 73–86.

[18] Passy, U., and Wilde, D.J., "Generalized polynomial optimization", *Journal on Applied Mathematics* 15/5 (1967) 1344–1356.

[19] Passy, U., "Generalized weighted mean programming", *SIAM Journal on Applied Mathematics* 20 (1971) 763–778.

[20] Phillips, D.T., and Beightler, C.S., "Geometric programming: A technical state-of-the-art survey", *AIIE Transactions* 5/2 (1973) 97–112.

[21] Phillips, D.T., and Reklaitis, G.V., "On geometric programming with slack constraints and degree of difficulty", *AIIE Transactions* 8/2 (1976) 275–279.

[22] Ratner, M., Lasdon, L.S., and Jain, A., "Solving geometric programs using GRG: Results and comparisons", *Journal of Optimization Theory and Applications* 26/2 (1978) 253–265.

[23] Reklaitis, G.V., and Wilde, D.J., "Geometric programming via primal auxiliary programs", *ORSA 38th National Meeting*, Dallas, Texas, May 1971.

[24] Rijckaert, M.J., and Martens, X.M., "Comparison of generalized geometric programming algorithms", *Journal of Optimization Theory and Applications* 26/2 (1978) 205–242.

[25] Smeers, Y., "Studies in geometric programming with applications to management science", Ph.D. Dissertation, Carnegie-Mellon University, 1972.

[26] Yang, H., "Investigation of path-following algorithms for signomial geometric programming problems", Ph.D. Dissertation, University of Iowa, 1994.