

USER'S GUIDE FOR TOMLAB /GP³

Marcus M. Edvall¹ and Fredrik Hellman²

November 6, 2006



¹Tomlab Optimization Inc., 855 Beech St #121, San Diego, CA, USA, medvall@tomopt.com.

²Tomlab Optimization AB, Västerås Technology Park, Trefasgatan 4, SE-721 30 Västerås, Sweden, fredrik@tomopt.com.

³Computational Optimization Laboratory, Department of Management Science, The University of Iowa, Iowa City, IA 52242, USA.
Computational Optimization Program Library: Geometric Programming.

Contents

Contents	2
1 Introduction	3
1.1 Overview	3
1.2 Contents of this Manual	3
1.3 More information	3
1.4 Prerequisites	3
2 Using the Matlab Interface	4
3 TOMLAB /GP Solver Reference	5
3.1 coplqpTL	5
4 Summary	8
5 Geometric Programming (GP)	8
6 The Algorithm	10
7 A GP Example	10
8 TOMLAB Test Set	11
9 Solution Output Files	14
References	17

1 Introduction

1.1 Overview

Welcome to the TOMLAB /GP (Geometric Programming) User's Guide. TOMLAB /GP includes the a geometric programming solver package and a convenient interface to The MathWorks' MATLAB.

The following sections describe the algorithm and TOMLAB format in more detail. There are several test problem included with the TOMLAB distribution that illustrates the use. The following example will create and run the first test case.

```
Prob = probInit('gp_prob', 1);  
Result = tomRun('GP', Prob, 1);
```

1.2 Contents of this Manual

- Section 4 provides a basic overview of the geometric programming solver.
- Section 5 describes geometric programming in detail.
- Section 6 gives algorithmic details.
- Section 7 illustrates how to solve a simple test case.
- Section 8 contains three examples to assist in the modeling of geometric programming problems.
- Section 9 shows the screen and file output.

1.3 More information

Please visit the following links for more information:

- <http://tomopt.com/tomlab/products/gp/>

1.4 Prerequisites

In this manual we assume that the user is familiar with linear/nonlinear programming, setting up problems in TOMLAB (in particular constrained nonlinear (**con**) problems) and the Matlab language in general.

2 Using the Matlab Interface

The GP solver is accessed via the *tomRun* driver routine, which calls the *coplgpTL* interface routine. The solver itself is located in the MEX file *coplgp.dll*.

Table 1: The interface routines.

Function	Description	Section	Page
<i>coplgpTL</i>	The interface routine called by the TOMLAB driver routine <i>tomRun</i> . This routine then calls the MEX file <i>gp</i>	3.1	5
<i>gpAssign</i>	The routine that creates a GP problem in the TOMLAB format.		

3 TOMLAB /GP Solver Reference

A detailed description of the TOMLAB /GP solver interface is given below. Also see the M-file help for *coplgpTL.m*.

3.1 coplgpTL

Purpose

Solve geometric programming problems as described by equation 7 and below.

Calling Syntax

```
Prob = gpAssign( ... );  
Result = tomRun('GP',Prob,...)
```

Description of Inputs

<i>Prob</i>	Problem description structure. The following fields are used:
<i>GP.A</i>	Linear constraint matrix for dual problem.
<i>GP.coef</i>	Posynomial coefficient vector.
<i>GP.nterm</i>	Number of terms in objective and constraints.
<i>GP.moremem</i>	Extra memory to allocate. Default $300*n + 300*m$, where n is the total number of terms and m is the number of variables. If this is not enough the interface will double the memory until it is enough and try to solve it. The final value of <i>moremem</i> will be returned in <i>Result.GP.lackmem</i> . If you solve many similar problems in a sequence, set <i>Prob.GP.moremem</i> to <i>Result.GP.lackmem</i> in order to avoid many unnecessary memory allocations.
<i>PriLevOpt</i>	Print level in the solver. See <i>GP.options.PRILEV</i> .
<i>GP</i>	Structure with GP solver specific fields.
<i>PrintFile</i>	Name of file to print progress information and results to. The output written to file is always with a print level of 2. Default: Empty, none that is.
<i>GP.options</i>	Structure with special fields for the GP solver:
<i>PRILEV</i>	Print level of GP solver. Has precedence over <i>Prob.PriLevOpt</i> . == 0 silent >= 1 normal output with iteration log >= 2 more verbose output >= 10 memory debug output >= 100 extreme debug output Default: 0

<i>Prob</i>	Problem description structure. The following fields are used:, continued
<i>ITLM</i>	Iteration limit. Default: 100
<i>TOLX</i>	Zero tolerance on the dual variables. Default: 1e-14
<i>TOLZ</i>	Zero tolerance on the constraint values. Default: 1e-14
<i>EPSP</i>	Feasibility tolerance on primal problem. Default: 1e-6
<i>EPSD</i>	Feasibility tolerance on dual problem. Default: 1e-6
<i>EPSC</i>	Feasibility tolerance on the linear dual constraints. Default: 1e-9
<i>TOLPIV</i>	Pivoting zero tolerance. Default: 1e-14
<i>FRAC</i>	Fractional decrease of steplength. Default: 0.9995
<i>BIG</i>	Value to treat as infinite. Default: 1e20

Description of Outputs

Result Structure with result from optimization. The following fields are set:

<i>f_k</i>	Function value at optimum.
<i>x_k</i>	Optimal point.
<i>xState</i>	State of each variable: 0/1/2/3: free / on lower bnd / on upper bnd / fixed.
<i>ExitFlag</i>	Exit status. The following values are used: 0: Optimal solution found. 1: Maximum number of iterations reached. 2: (Possibly) unbounded problem. 4: (Possibly) infeasible problem.
<i>ExitText</i>	The exit text according to Inform.
<i>Inform</i>	Status value returned from the GP solver. 0: Optimal: find an optimal solution. 1: Unbounded: the reformulated problem is unbounded. 2: Infeasible: the reformulated problem is infeasible. 3: Iteration limit: reach the iteration limit before desired accuracy. 4: Numerical difficulties: encounter numerical problems. 5: Primal or dual infeasible: the original problems are infeasible. 6: Insufficient space: memory space is not sufficient. 7: Data file error: find input data errors.
<i>Solver</i>	Name of the solver (GP).
<i>SolverAlgorithm</i>	Description of the solver.
<i>GP.lackmem</i>	See description for input Prob.GP.moremem.

4 Summary

COPL_GP (Computational Optimization Program Library: Geometric Programming) is a highly efficient and accurate computer program developed in Computational Optimization Laboratory, Department of Management Science, The University of Iowa, Iowa City, IA 52242, USA (<http://dollar.biz.uiowa.edu/col/ye>) for solving the classical posynomial geometric programming problems. The approach is by means of a primal–dual algorithm developed simultaneously for (i), the dual geometric program after logarithmic transformation of its objective function and (ii), its Lagrangian dual program. Under rather general assumptions, the mechanism defines a primal–dual infeasible path from a specially constructed, perturbed Karush–Kuhn–Tucker system. *Subfeasible* solutions, are generated for each program whose primal and dual objective function values converge to the respective primal and dual program values.

The basic technique is one of a predictor–corrector type involving Newton’s method applied to the perturbed KKT system, coupled with effective techniques for choosing iterate directions and step lengths. Sophisticated implementation techniques and advanced sparse matrix factorizations are used to take advantage of the very special structure of the Hessian matrix of the logarithmically transformed dual objective function.

Computational results on 19 of the most challenging GP problems found in the literature are encouraging. The performance indicates that the algorithm is effective regardless of the *degree of difficulty*, which is a generally accepted measure in geometric programming.

Geometric programming GP is a very broad class of mathematical problems which is useful in the study of a variety of optimization problems. Its great impact has been in the areas of (1) **engineering design** : [8],[2], [11], and [12]; (2) **economics & statistics** : [5], [17], [4], [13],[1], and [3]; (3) **manufacturing**: [14], [10],[18], and (4) **chemical equilibrium** : [8], [2], [16]).

The technical details of COPL_GP can be found in Kortanek, Xu and Ye [15]

5 Geometric Programming (GP)

The *primal GP* is :

$$\begin{aligned}
 (GP) \quad V_{GP} := \quad & \text{minimize} \quad g_0(t) \\
 & \text{subject to} \quad g_k(t) \leq 1, \quad k = 1, 2, \dots, p \\
 & \quad \quad \quad t_i > 0, \quad i = 1, 2, \dots, m
 \end{aligned} \tag{1}$$

where

$$g_0(t) = \sum_{j=1}^{n_0} c_j t_1^{a_{1j}} \dots t_m^{a_{mj}} \tag{2}$$

$$g_k(t) = \sum_{j=n_{k-1}+1}^{n_k} c_j t_1^{a_{1j}} \dots t_m^{a_{mj}}, \quad k = 1, 2, \dots, p. \tag{3}$$

Given exponents a_{ij} for the i th variable in the j th product term, $i = 1, \dots, m$ and $j = 1, \dots, n_p$, are arbitrary real constants and term coefficients c_j are positive.

Here, $g_0(t)$ is called the objective function and $g_k(t)$ the k th constraint function, where n_0 is the number of product terms in the objective function and $(n_k - n_{k-1})$ is the number of product terms in the k constraint

function. Therefore, the number of the total product terms is n_p , where p is the number of constraint functions. The vector t contains m variables, denoted by t_1, \dots, t_m .

The program solves *posynomial GP*, and so we shall remove *posynomial* with this understanding.

The dual to *GP* ([8]) is

$$\begin{aligned}
(GD) \quad V_{GD} := \quad & \text{maximize} \quad \prod_{j=1}^{n_p} (c_j/x_j)^{x_j} \prod_{k=1}^p \lambda_k^{\lambda_k} \\
& \text{subject to} \quad \sum_{j=1}^{n_0} x_j = 1, \\
& \sum_{j=1}^{n_p} x_j a_{ij} = 0, \quad i = 1, 2, \dots, m \\
& x_j \geq 0, \quad j = 1, 2, \dots, n_p
\end{aligned} \tag{4}$$

where

$$\lambda_k = \sum_{j=n_{k-1}+1}^{n_k} x_j, \quad k = 1, 2, \dots, p. \tag{5}$$

For a (primal) *GP* having m variables (t_i), p constraints and n_p (posynomial) product terms we see that (dual) *GD* has n_p non-negative variables (x_j) in $m+1$ linear equations. In the literature the *degree of difficulty* of a *GP* is defined by :

$$\text{degree of difficulty} = n_p - m - 1. \tag{6}$$

Let $F(x)$ denote the negative of the logarithm of the objective function of (*GD*), i.e.,

$$F(x) = \sum_{j=1}^{n_0} x_j \ln\left(\frac{x_j}{c_j}\right) + \sum_{k=1}^p \sum_{j=n_{k-1}+1}^{n_k} x_j \ln\left(\frac{x_j}{c_j \lambda_k}\right). \tag{7}$$

We see that (*GD*) is a linearly constrained convex programming problem,

$$\begin{aligned}
& \text{minimize} \quad F(x) \\
& \text{subject to} \quad Ax = b, \quad x \geq 0,
\end{aligned}$$

where the coefficient matrix is given by

$$A = \begin{pmatrix} 1 & \dots & 1 & 0 & \dots & 0 & \dots & 0 & \dots & 0 \\ a_{1,1} & \dots & a_{1,n_0} & a_{1,n_0+1} & \dots & a_{1,n_1} & \dots & a_{1,n_{p-1}+1} & \dots & a_{1,n_p} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{m,1} & \dots & a_{m,n_0} & a_{m,n_0+1} & \dots & a_{m,n_1} & \dots & a_{m,n_{p-1}+1} & \dots & a_{m,n_p} \end{pmatrix}, \tag{8}$$

and right side hand

$$b^T = (1, 0, \dots, 0) \in R^{m+1}. \tag{9}$$

Note that the first n_0 entries of the first row of A are ones, and data a_{ij} are stored from row 2 to row $m+1$ in A .

6 The Algorithm

The program is an implementation of an interior–point algorithm for linear constrained convex programming which when applied to geometric programming solves both primal and dual *GP* problems simultaneously. Using the perturbed KKT system of the dual *GP*, a central–path is defined which converges to a solution of both the primal and dual *GP* programs. The algorithm does not require the existence of an interior point for either program. Moreover, the algorithm has the feature of generating *subfeasible solutions*, [9, 8], when the primal geometric program is inconsistent but subconsistent. Computing functional values of subfeasible solutions gives rise to more conceivable duality states for the primal–dual pair. In this manual we limit these possibilities by considering “boundedly subfeasible” sequences for which our algorithm delivers a path whose primal–dual gap can be made arbitrarily small, even for the case where one of the pair of dual problem is inconsistent. **In addition, the objective functions need not be differentiable at an optimal solution of a given consistent program.**

Each iteration of the algorithm computes the Newton direction from a perturbed KKT system involving two parameters, γ and η . The affine direction (corresponding to a particular setting $\gamma = 0, \eta = 1$) is used to predict reductions in both the feasibility and complementarity residuals. Then proper parameters γ, η are chosen for the KKT system which is solved for the direction again. The indefinite reduced KKT matrix is factorized by performing diagonal pivots whose orders are chosen in terms of minimum degree to maintain the sparsity of the factors. A static data structure associated with the chosen pivoting order is allocated by a single symbolic factorization and used in subsequent iterations. Tactics for taking large steps and reset dual slack variables according to certain rules are implemented.

Computational results indicate that the algorithm, coupled with these new techniques, leads to an efficient and stable implementation for solving primal and dual geometric programs simultaneously. The results indicate that the standard geometric programming measure of difficulty is no barrier to the methods.

7 A GP Example

This example is called Dembo78, [7]:

$$(DP) \quad \min \{t_1 t_2 + t_1^{-1} t_2^{-1} \mid (1/4)t_1^{1/2} + t_2 \leq 1, t_1 > 0, t_2 > 0\}.$$

The *GP* dual to Dembo78 has a unique optimal solution, $x = (0.5, 0.5, 0, 0)$:

$$(DD) \quad \begin{array}{ll} \max & (1/x_1)^{x_1} (1/x_2)^{x_2} (0.25/x_3)^{x_3} (1/x_4)^{x_4} (x_3 + x_4)^{(x_3+x_4)} \\ \text{subject to} & x_1 + x_2 = 1 \\ & x_1 - x_2 + 0.5x_3 = 0 \\ & x_1 - x_2 + x_4 = 0 \\ & x \geq 0 \end{array}$$

The TOMLAB commands are:

```
Prob = gpAssign([2; 2], [1 1 1/4 1], ...
               [1 -1 0.5 0; 1 -1 0 1]');

```

```
Result = tomRun('gp', Prob, 1);

```

The problem is also defined in *gp_prob*:

```
Prob = probInit('gp_prob', 1);  
  
Result = tomRun('gp', Prob, 1);
```

8 TOMLAB Test Set

There are several test problems included with the TOMLAB distribution. Three of the problems are illustrated below (P1, P4 and P10). These problems are created and solved by executing:

```
Prob1 = probInit('gp_prob', 3); % Create P1.  
Prob2 = probInit('gp_prob', 6); % Create P4.  
Prob3 = probInit('gp_prob', 12); % Create P10A.  
  
Result1 = tomRun('GP', Prob1, 2); % Solve P1.  
Result2 = tomRun('GP', Prob2, ,2); % Solve P4.  
Result3 = tomRun('GP', Prob3, ,2); % Solve P10A.
```

Problem P1:

$$\begin{aligned} (P1) \quad \min \quad & 5x_1 + 50000x_1^{-1} + 20x_2 + 72000x_2^{-1} + 10x_3 + 144000x_3^{-1} \\ \text{subject to} \quad & 4x_1^{-1} + 32x_2^{-1} + 120x_3^{-1} \leq 1 \\ & x \geq 0 \end{aligned}$$

Using TOMLAB the problem is modeled as:

Problem P1 in TOMLAB:

$$A = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & -1 \end{pmatrix}^T \quad (10)$$

$$coef = (5 \quad 50000 \quad 20 \quad 72000 \quad 10 \quad 144000 \quad 4 \quad 32 \quad 120)^T \quad (11)$$

$$nterm = (6 \quad 3)^T \quad (12)$$

```
Prob = gpAssign(nterm, coef, A, 'P1');
```

```
Result = tomRun('GP', Prob, ,2);
```

similarly for the other test cases:

Problem P4:

$$\begin{aligned} (P1) \quad & \min \quad x_1^{-1} x_2^{-1} x_3^{-1} \\ & \text{subject to} \quad 2x_1 + x_2 + 3x_3 \leq 1 \\ & \quad \quad \quad x_1 + 3x_2 + 2x_3 \leq 1 \\ & \quad \quad \quad x_1 + x_2 + x_3 \leq 1 \\ & \quad \quad \quad x \geq 0 \end{aligned}$$

Problem P4 in TOMLAB:

$$A = \begin{pmatrix} -1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}^T \quad (13)$$

$$coef = (1 \ 2 \ 1 \ 3 \ 1 \ 3 \ 2 \ 1 \ 1 \ 1)^T \quad (14)$$

$$nterm = (1 \ 3 \ 3 \ 3)^T \quad (15)$$

```
Prob = gpAssign(nterm, coef, A, 'P4');
```

```
Result = tomRun('GP', Prob, ,2);
```

Problem P10A:

$$\begin{aligned}
(P1) \quad \min \quad & 2x_1^{0.9}x_2^{-1.5}x_3^{-3} + 5x_4^{-0.3}x_5^{2.6} + 4.7x_6^{-1.8}x_7^{-0.5}x_8 \\
\text{subject to} \quad & 7.2x_1^{-3.8}x_2^{2.2}x_3^{4.3} + 0.5x_4^{-0.7}x_5^{-1.6} + 0.2x_6^{4.3}x_7^{-1.9}x_8^{8.5} \leq 1 \\
& 10x_1^{2.3}x_2^{1.7}x_3^{4.5} \leq 1 \\
& 0.6x_4^{-2.1}x_5^{0.4} \leq 1 \\
& 6.2x_6^{4.5}x_7^{-2.7}x_8^{-0.6} \leq 1 \\
& 3.1x_1^{1.6}x_2^{0.4}x_3^{-3.8} \leq 1 \\
& 3.7x_4^{5.4}x_5^{1.3} \leq 1 \\
& 0.3x_6^{-1.1}x_7^{7.3}x_8^{-5.6} \leq 1 \\
& x \geq 0
\end{aligned}$$

Problem P10A in TOMLAB:

$$A = \begin{pmatrix} 0.9 & 0 & 0 & -3.8 & 0 & 0 & 2.3 & 0 & 0 & 1.6 \\ 0 & 0 & & & & & & & & \\ -1.5 & 0 & 0 & 2.2 & 0 & 0 & 1.7 & 0 & 0 & 0.4 \\ 0 & 0 & & & & & & & & \\ -3 & 0 & 0 & 4.3 & 0 & 0 & 4.5 & -2.1 & 0 & -3.8 \\ 0 & 0 & & & & & & & & \\ 0 & -0.3 & 0 & 0 & -0.7 & 0 & 0 & -2.1 & 0 & 0 \\ 5.4 & 0 & & & & & & & & \\ 0 & 2.6 & 0 & 0 & -1.6 & 0 & 0 & 0.4 & 0 & 0 \\ 1.3 & 0 & & & & & & & & \\ 0 & 0 & -1.8 & 0 & 0 & 4.3 & 0 & 0 & 4.5 & 0 \\ 0 & -1.1 & & & & & & & & \\ 0 & 0 & -0.5 & 0 & 0 & -1.9 & 0 & 0 & -2.7 & 0 \\ 0 & 7.3 & & & & & & & & \\ 0 & 0 & 1 & 0 & 0 & 8.5 & 0 & 0 & -0.6 & 0 \\ 0 & -5.6 & & & & & & & & \end{pmatrix}^T \tag{16}$$

$$coef = \begin{pmatrix} 2 & 5 & 4.7 & 7.2 & 0.5 & 0.2 & 10 & 0.6 & 6.2 & 3.1 \\ 3.7 & 0.3 & & & & & & & & \end{pmatrix}^T \tag{17}$$

$$nterm = (3 \ 3 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1)^T \tag{18}$$

`Prob = gpAssign(nterm, coef, A, 'P10A');`

```
Result = tomRun('GP', Prob, ,2);
```

9 Solution Output Files

The output file contains three sections.

- The first section “geometric programming model” describes some problem characteristics, such as the number of variables, etc.
- The second section “path-following solver for reformulated model” contains the running information during the iterative process. The primal and dual objective values represent the reformulated primal and dual problems.
- The third section “solutions for original GP” contains the optimal values of the variables, the constraint functions, the primal and dual objective functions, and the feasibility residues of the original GP problem pair.

The termination code 0 – –7 represents the following:

```
0=optimal: find an optimal solution;
1=unbounded: the reformulated problem is unbounded;
2=infeasible: the reformulated problem is infeasible;
3=iteration limit: reach the iteration limit before desired accuracy;
4=numerical difficulties: encounter numerical problems
5=primal or dual infeasible: the original problems are infeasible;
6=insufficient space: memory space is not sufficient;
7=data file error: find input data errors.
```

The last paragraph lists the CPU time: read in data, solve the problem and the sum of the two.

The following is the output file for Dembo68.

```
=====
COPL_GP 1.1 2005
=====
geometric programming optimizer

geometric programming model
-----
primal gp vars . 2
primal gp consts 1
primal gp terms 4
degree of diffic 1
```

path-following solver for reformulated model

```
-----  
*- iter = 0      dual value      =-2.00000000000000E+00  
                 primal value     =-4.64038529823796E-17  
   mu = 9.233E-01 Dual infeasibility = 3.608E-01  
                 Primal infeasibility = 5.000E-01  
  
*- iter = 1      dual value      =-9.99533217451737E-01  
                 primal value     =-3.62989378647483E-01  
   mu = 1.591E-01 Dual infeasibility = 0.000E+00  
                 Primal infeasibility = 1.872E-17  
  
*- iter = 2      dual value      =-7.13113168002422E-01  
                 primal value     =-5.88706966976516E-01  
   mu = 3.544E-02 Dual infeasibility = 2.294E-02  
                 Primal infeasibility = 8.301E-17  
  
*- iter = 3      dual value      =-6.93636390346386E-01  
                 primal value     =-6.89738422619333E-01  
   mu = 1.854E-03 Dual infeasibility = 4.382E-03  
                 Primal infeasibility = 2.975E-17  
  
*- iter = 4      dual value      =-6.93130105030368E-01  
                 primal value     =-6.93115339650404E-01  
   mu = 9.392E-06 Dual infeasibility = 2.847E-05  
                 Primal infeasibility = 2.383E-17  
  
*- iter = 5      dual value      =-6.93147171812859E-01  
                 primal value     =-6.93147164639831E-01  
   mu = 4.710E-09 Dual infeasibility = 1.456E-08  
                 Primal infeasibility = 4.592E-17  
  
*- iter = 6      dual value      =-6.93147180555572E-01  
                 primal value     =-6.93147180551985E-01  
   mu = 2.355E-12 Dual infeasibility = 7.282E-12  
                 Primal infeasibility = 8.860E-17  
  
*- iter = 7      dual value      =-6.93147180559943E-01  
                 primal value     =-6.93147180559941E-01  
   mu = 1.210E-15 Dual infeasibility = 3.705E-15  
                 Primal infeasibility = 5.133E-17
```

-- exit : optimal solution obtained. iter 7

```
    dual objective value =-6.93147180559943E-01  
    primal                =-6.93147180559941E-01
```

```
Dual infeasibility = 3.705E-15
Primal             = 5.133E-17
```

```
solutions for original GP
```

```
-----
primal GP solution
```

```
   i           t(i)
   1  .421974819510E+01
   2  .236980965158E+00
```

```
primal GP constraint value
```

```
   i  constraint value
   1  .750531607446E+00
```

```
primal GP obj value (from direct comp)  2.000000000000000E+00
primal GP infeasibility                  0.000E+00
```

```
dual   GP obj value (from direct comp)  1.999999999999999E+00
dual   GP infeasibility                  5.133E-17
```

```
all done, termination code= 0
0=optimal, 1=unbounded, 2=infeasible
3=iteration limit, 4=numerical difficulties
5=primal or dual infeasible, 6=insufficient space
7=data file error
```

```
----- time for readin = .02
           solving = .01
total time in seconds = .03
```

Tolerances for primal and dual feasibilities and complementarity are specified as

$$\epsilon_P = 10^{-8}, \quad \epsilon_D = 10^{-8}, \quad \text{and} \quad \epsilon_C = 10^{-12}. \quad (19)$$

Extensive computational tests indicate that the requirement of $\epsilon_C = 10^{-12}$ is very strong. The solution stopping by this specification usually is very accurate.

References

- [1] H. El Barmi, and R.L. Dykstra, "Restricted multinomial maximum likelihood estimation based upon Fenchel duality," *Statistics and Probability Letters* 21 (1994) 121-130.
- [2] C.S. Beightler and D.T. Phillips, *Applied Geometric Programming* (John Wiley & Sons, NY, 1976).
- [3] D.L. Bricker, K.O. Kortanek and L. Xu, "Maximum likelihood estimates with order restrictions on probabilities and odds ratios: a geometric programming approach", Applied Mathematical and Computational Sciences, The University of IA, Iowa City, Iowa 52242, September, 1995
- [4] A. Charnes, W.W. Cooper, B. Golany and J. Masters, "Optimal design modification by geometric programming and constrained stochastic network models," *International Journal on Systems Science* 19 (1988) 825-844.
- [5] A. Charnes, W.W. Cooper and K.O. Kortanek, "Semi-infinite programming, differentiability and geometric Programming," *Journal of Mathematical Sciences* 6 (1971) 19-40.
- [6] R.S. Dembo, "A set of geometric programming test problems and their solutions," *Mathematical Programming* 10 (1976) 192-213.
- [7] R.S. Dembo, "Dual to primal conversion in geometric programming," *Journal of Optimization Theory and Applications* 26 (1978) 243-252.
- [8] R.J. Duffin, E.L. Peterson and C. Zener, *Geometric Programming-Theory and Application* (John Wiley & Sons, NY, 1967).
- [9] R.J. Duffin, "Infinite Programs", in: H. W. Kuhn and A. W. Tucker, eds., *Linear Inequalities and Related Systems*, (Princeton University Press, Princeton, NJ, 1956) pp.157-170
- [10] J. Dupacová, P. Charamza and J. Mádl "On stochastic aspects of a metal cutting problem," Department of Statistics, Charles University, Sokolovská 83, 18600 Prague, Czech Republic, June, 1993.
- [11] A.V. Fiacco and A. Ghaemi, "Sensitivity analysis of a nonlinear water pollution control model using an upper hudson river data base," The George Washington University, Washington, D.C., August, 1981.
- [12] A.V. Fiacco and A. Ghaemi, "Sensitivity and parametric bound analysis of an electric power generator GP model: optimal steam turbine-exhaust annulus and condenser sizes, serial T-437," The George Washington University, Washington, D.C., August, 1981
- [13] R. Jagannathan, "A stochastic geometric programming problem with multiplicative recourse," *Operations Research Letters* 9 (1990) 99-104.
- [14] S. Jha, K.O. Kortanek and H. No, "Lotsizing and setup time reduction under stochastic demand: a geometric programming approach," Working paper series No. 88-12, College of Business Administration, The University of Iowa, Iowa City, June, 1988
- [15] K.O. Kortanek, X. Xu and Y. Ye, "An infeasible interior-point algorithm for solving primal and dual geometric programs," *Mathematical Programming* 76 (1996) 155-181.
- [16] M. J. Rijckaert and X. M. Martens, "Bibliographical note on geometric programming," *Journal of Optimization Theory and Applications* 26 (1978) 325-337

- [17] T. Robertson, F.T. Wright and R.L. Dykstra, *Order Restricted Statistical Inference* (John Wiley and Sons, NY, 1988).
- [18] C.H. Scott and T.R. Jefferson, "Allocation of resources in project management," *International Journal on Systems Science* 26 (1995) 413-420.