Stochastic geometric optimization with joint probabilistic constraints

Jia Liu[a,b], Abdel Lisser [1a], Zhiping Chen[b]

[a]*Laboratoire de Recherche en Informatique (LRI), Université Paris Sud - XI, Bât. 650, 91405 Orsay Cedex, France*
[b]*School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an, Shaanxi, 710049, P. R. China*

Abstract

This paper discusses geometric programs with joint probabilistic constraints. When the stochastic parameters are normally distributed and independent of each other, we approximate the problem by using piecewise polynomial functions with non-negative coefficients, and transform the approximation problem into a convex geometric program. We prove that this approximation method provides a lower bound. Then, we use an improved Bonferroni approximation method to find an upper bound. Finally, numerical tests are carried out with a shape optimization problem.

Keywords: Geometric program, Joint probabilistic constraint, Piecewise polynomial approximation, Bonferroni approximation.

1. Introduction

Geometric programs are a type of optimization problems characterized by an objective and constraints functions which have a special form. A number of practical problems, such as electrical circuit design problems [1], mechanical engineering problems [12], economic and managerial problems [8] and nonlinear network problems [7], can be formulated by geometric programs.

[1]Corresponding author.
E-mail addresses: liu.jia@stu.xjtu.edu.cn (J.Liu), lisser@lri.fr (A.Lisser), zchen@xjtu.edu.cn (Z.Chen).

A geometric program can be formulated as

$$\min_t g_0(t) \text{ s.t. } g_k(t) \leq 1,\ k = 1, \cdots, K,\ t \in \mathbb{R}^M_{++} \tag{1}$$

with

$$g_k(t) = \sum_{i \in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}},\ k = 0, \cdots, K. \tag{2}$$

Usually, $c_i \prod_{j=1}^{M} t_j^{a_{ij}}$ is called a monomial, and $g_k(t)$ is called posynomial. We denote Q the number of monomials in (1), and $\{I_k,\ k = 0, \cdots, K\}$ is the disjoint index sets of $\{1, \cdots, Q\}$.

Geometric programs are not convex with respect to t whilst they are convex with respect to $\{z : z_j = \log\ t_j,\ j = 1, \cdots, M\}$. Hence, interior point method can be efficiently used to solve geometric programs.

In real world applications, some of the coefficients in (1) may not be known precisely when the optimization is made. Hence, the stochastic geometric programming is proposed to model geometric problems with random parameters. Individual probabilistic constraints have been used to control the uncertainty level of the constraints in (1):

$$P(\sum_{i \in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \leq 1) \geq 1 - \epsilon_k,\ k = 1, \cdots, K, \tag{3}$$

where ϵ_k is the tolerance probability for the k-th constraint in (2). It is shown in [3, 6, 11] that when the coefficients $a_{ij}, i \in I_k, \forall k, j = 1, \ldots, M$, are deterministic and $c_i, i \in I_k, \forall k$ are normally distributed and independent of each other, the probabilistic constraint (3) is equivalent to two constraints involving posynomials and common additional slack variables.

2. Stochastic geometric programs with joint probabilistic constraints

In this paper, we consider the following joint probabilistic constrained stochastic geometric programs

$$\min_t \quad E\left[\sum_{i \in I_0} c_i \prod_{j=1}^{M} t_j^{a_{ij}}\right] \tag{4}$$

$$\text{s.t.} \quad P\left(\sum_{i \in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \leq 1,\ k = 1, \cdots, K\right) \geq 1 - \epsilon. \tag{5}$$

Unlike [3, 6, 11], we require that the overall probability of meeting the K geometric constraints is above a certain probability level $1-\epsilon$, where $\epsilon \in (0, 0.5]$.

Stochastic geometric programs with joint probabilistic constraints are a special case of joint probabilistic constrained problems. The latter were first considered by Miller and Wagner [9]. Under some independence assumptions, they show that joint probabilistic constrained problems are equivalent to concave deterministic problems. For some specific cases, such as the right hand side random vector is multivariate normally distributed, Prekopa [10] showed that the joint probabilistic constraint problems are convex. Moreover, some approaches are proposed for linear programs with joint probabilistic constraints in [2] and the references therein. However, as far as we know, there is no in-depth research works on the stochastic geometric programs with joint probabilistic constraints. Hence, in this paper, we propose new approaches for solving this problem under normal distribution with independent components.

2.1. Stochastic geometric optimization under Gaussian distribution

We suppose that the coefficients $a_{ij}, i \in I_k, \forall k, j = 1, \ldots, M$, are deterministic and the parameters $c_i, i \in I_k, \forall k$ are normally distributed and independent of each other, i.e., $c_i \sim N(E_{c_i}, \sigma_i^2)$ [3]. Moreover, we assume that $E_{c_i} \geq 0$. As c_i are independent of each other, constraint (5) is equivalent to

$$\prod_{k=1}^{K} P(\sum_{i \in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \leq 1) \geq 1-\epsilon. \tag{6}$$

By introducing auxiliary variables $y_k \in \mathbb{R}_+,\ k = 1, \cdots, K$, (6) can be equivalently transformed into

$$P(\textstyle\sum_{i \in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \leq 1) \geq y_k,\ k = 1, \cdots, K, \tag{7}$$

and

$$\textstyle\prod_{k=1}^{K} y_k \geq 1-\epsilon,\ 1 \geq y_k \geq 0,\ k = 1, \cdots, K. \tag{8}$$

It is easy to see that for independent normally distributed $c_i \sim N(E_{c_i}, \sigma_i^2)$ [3], constraint (7) is equivalent to

$$\textstyle\sum_{i \in I_k} E_{c_i} \prod_{j=1}^{M} t_j^{a_{ij}} + \Phi^{-1}(y_k) \sqrt{\sum_{i \in I_k} \sigma_i^2 \prod_{j=1}^{M} t_j^{2a_{ij}}} \leq 1,\ k = 1, \cdots, K. \tag{9}$$

Here, $\Phi^{-1}(y_k)$ is the quantile of the standard normal distribution $N(0,1)$. However, biconvex inequalities (9) are still very hard to solve within an optimization problem [4].

2.2. Approximation of $\Phi^{-1}(\cdot)$

$\Phi^{-1}(\cdot)$ is also called the probit function, and it can be expressed in terms of the inverse error function:

$$\Phi^{-1}(y_k) = \sqrt{2}\,\mathrm{erf}^{-1}(2y_k - 1), \quad y_k \in (0,1).$$

The inverse error function is a nonelementary function which can be represented by the Maclaurin series:

$$\mathrm{erf}^{-1}(z) = \sum_{p=0}^{\infty} \frac{\lambda_p}{2p+1} \left(\frac{\sqrt{\pi}}{2} z \right)^{2p+1},$$

where $\lambda_0 = 1$ and

$$\lambda_p = \sum_{i=0}^{p-1} \frac{\lambda_i \lambda_{p-1-i}}{(i+1)(2i+1)} > 0, \ p = 1, 2, \cdots.$$

Thus, we know that $\Phi^{-1}(y_k)$ is convex for $1 \geq y_k \geq 0.5$, and concave for $0 \leq y_k \leq 0.5$.

From constraint (8), we have $0.5 \leq 1 - \epsilon \leq y_k \leq 1$. Hence, we can only focus on the right tail part of $\Phi^{-1}(y_k)$. For the right tail part, we use polynomial functions with non-negative coefficients to construct convex approximations of $\Phi^{-1}(y_k)$.

In detail, we choose S polynomials:

$$F_s(y_k) = \sum_{p=1}^{n_s} \beta_{s,p} y_k^p + \beta_{s,0}, \ s = 1, \cdots, S,$$

such that

$$\beta_{s,p} \geq 0, \ s = 1, \cdots, S, \ p = 0, 1, \cdots, n_s, \tag{10}$$

and

$$F_s(y_k) \leq \Phi^{-1}(y_k), \ \forall y_k \in [1-\epsilon, 1), \ s = 1, \cdots, S, \tag{11}$$

where n_s is the degree of the s^{th} polynomial. $\Phi^{-1}(y_k)$ is then approximated by a piecewise polynomial function

$$F(y_k) = \max_{s=1,\cdots,S} F_s(y_k). \tag{12}$$

Constraints (11) guarantee that $F(y_k)$ provides a lower bound of $\Phi^{-1}(y_k)$. Moreover, constraints (10) are used to keep the geometric programming structure of the approximation problem.

In order to guarantee the strength of the approximation, we use an optimization model to compute the optimal coefficients $\beta_{s,i}$ such that the bias of the polynomial $F_s(y_k)$ from $\Phi^{-1}(y_k)$ is minimal. For given s, we first choose $\hat{M}$ points, $\xi_1, \xi_2, \cdots, \xi_{\hat{M}}$, in the interval $[1-\epsilon, 1)$, randomly or uniformly. We then minimize the sum of the differences between $F_s(\xi_m)$ and $\Phi^{-1}(\xi_m)$ on the $\hat{M}$ points such that (10) and (11) hold:

$$\min_{\beta_{s,p}} \quad \sum_{m=1}^{\hat{M}} \left(\Phi^{-1}(\xi_m) - F_s(\xi_m)\right) \tag{13}$$

$$\text{s.t.} \quad F_s(y_k) \le \Phi^{-1}(y_k), \ \forall y_k \in [1-\epsilon, 1), \tag{14}$$

$$\beta_{s,p} \ge 0, \ p = 0, 1, \cdots, n_s. \tag{15}$$

The optimization problem (13)-(15) is a semi-infinite programming problem, which is very hard to solve directly. In order to guarantee the feasibility of constraints (14), we can not approximate the semi-infinite constraint by finite constraints on the $\hat{M}$ points.

An important property of $\Phi^{-1}(y_k)$ is that its derivative of any order q, $(\Phi^{-1})^{(q)}(y_k)$, is larger than or equal to 0 for $y_k \in [0.5, 1)$, and $(\Phi^{-1})^{(q)}(1) = +\infty$, for $q = 0, 1, \cdots$. From this property, we propose an approximation for the semi-infinite constraint (14).

Proposition 1. *A sufficient condition for constraints (14) is*

$$F_s^{(q)}(1-\epsilon) \le (\Phi^{-1})^{(q)}(1-\epsilon), \ q = 0, 1, \cdots, n_s, \tag{16}$$

where $F_s^{(q)}$ and $(\Phi^{-1})^{(q)}$ are the the q^{th} derivative function of F_s and (Φ^{-1}).

Proof. Let $f(x) = \Phi^{-1}(x) - F_s(x)$. From (16), we have $f^{(q)}(1-\epsilon) \ge 0$, $q = 0, 1, \cdots, n_s$. We first consider the q^{th} derivative of $f(x)$ for $x \in [1-\epsilon, 1)$. From the Maclaurin series of $\Phi^{-1}(x)$, we know that $(\Phi^{-1})^{(n_s+1)}(x) \ge 0$, $\forall x \in$

$[1-\epsilon, 1)$ whilst $F_s^{(n_s)}(x) = \beta_{s,n_s} p!$ and $F_s^{(n_s+1)}(x) = 0$, $\forall x \in [1-\epsilon, 1)$. Then, we have $f^{(n_s+1)}(x) \geq 0$, $\forall x \in [1-\epsilon, 1)$. This means $f^{(n_s)}(x)$ is monotone increasing in $[1-\epsilon, 1)$. Moreover, we have $f^{(n_s)}(x) \geq f^{(n_s)}(1-\epsilon) \geq 0$, $\forall x \in [1-\epsilon, 1)$. Furthermore, the non-negativeness of $f^{(n_s)}(x)$ in $[1-\epsilon, 1)$ implies the monotonicity and non-negativeness of $f^{(n_s-1)}(x)$. By doing the above argument recursively from $f^{(n_s)}(x)$ to $f(x)$, we have $f(x) \geq 0$, $\forall x \in [1-\epsilon, 1)$, which is the conclusion of the proposition. □

Using (16) instead of (14), we find a feasible approximation for the semi-infinite problem (13-15). It can be formulated as the following linear programming problem

$$\min_{\beta_{s,p}} \quad \sum_{m=1}^{\hat{M}} \left(\Phi^{-1}(\xi_m) - \sum_{p=0}^{n_s} \xi_m^p \beta_{s,p} \right) \tag{17}$$

$$\text{s.t.} \quad \sum_{p=q}^{n_s} \frac{p!}{(p-q)!} (1-\epsilon)^{p-q} \beta_{s,p} \leq (\Phi^{-1})^{(q)}(1-\epsilon), \ q = 0, 1, \cdots, n_s, \tag{18}$$

$$\beta_{s,p} \geq 0, \ p = 0, 1, \cdots, n_s, \tag{19}$$

2.3. Convex geometric approximation

After obtaining the piecewise polynomial approximation of $\Phi^{-1}(\cdot)$, we can replace $\Phi^{-1}(y_k)$ by $F(y_k)$ in (9), and then we have

Theorem 1. *Using the piecewise polynomial function $F(y_k)$, we have the following approximation of the stochastic geometric program with a joint probabilistic constraint (4)-(5):*

$$\min_{t,y} \quad \sum_{i \in I_0} E_{c_i} \prod_{j=1}^{M} t_j^{a_{ij}} \tag{20}$$

$$\text{s.t.} \quad \sum_{i \in I_k} E_{c_i} \prod_{j=1}^{M} t_j^{a_{ij}} + \left(\sum_{p=0}^{n_s} \beta_{s,p} y_k^p \right) \sqrt{\sum_{i \in I_k} \sigma_i^2 \prod_{j=1}^{M} t_j^{2a_{ij}}} \leq 1,$$

$$s = 1, \cdots, S, \ k = 1, \cdots, K, \tag{21}$$

$$\prod_{k=1}^{K} y_k \geq 1-\epsilon, \ 1 \geq y_k \geq 0, \ k = 1, \cdots, K. \tag{22}$$

The optimal value of the approximation problem (20)-(22) is a lower bound of the problem (4)-(5).

We call this approximation as polynomial approximation.

Proof. From (7) and (8), we know $0 \leq y_k \leq 1$. Moreover, we have from (8) that for any k, $y_k \geq \prod_{k=1}^{K} y_k \geq 1 - \epsilon \geq 0.5$. Then, from (11), we have

$$\Phi^{-1}(y_k) \geq \sum_{p=0}^{n_s} \beta_{s,p} y_k^p, \ \forall y_k \in [1-\epsilon, 1), \ s = 1, \cdots, S.$$

Furthermore, constraint (21) is equivalent to

$$\sum_{i \in I_k} E_{c_i} \prod_{j=1}^{M} t_j^{a_{ij}} + \max_{s=1,\cdots,S} \left(\sum_{p=0}^{n_s} \beta_{s,p} y_k^p \right) \sqrt{\sum_{i \in I_k} \sigma_i^2 \prod_{j=1}^{M} t_j^{2a_{ij}}} \leq 1, \ k = 1, \cdots, K.$$

As $\sum_{p=0}^{n_s} \beta_{s,p} y_k^p \leq \Phi^{-1}(y_k)$, $\forall s$, any feasible solution for (9) is feasible for (21). From the equivalence between (5) and (9) under the Guassian distribution assumption, the optimal solution of problem (4)-(5) is feasible for problem (20)-(22). This means that the approximation problem (20)-(22) provides a lower bound for the original problem (4)-(5).

□

Although problem (20)-(22) is biconvex with respect to t and y, we can transform it into the following optimization problem by letting $r_j = \log(t_j), \ j = 1, \cdots, M$ and $x_k = \log(y_k), \ k = 1, \cdots, K$:

$$\min_{r,x} \quad \sum_{i \in I_0} E_{c_i} \exp\left\{ \sum_{j=1}^{M} a_{ij} r_j \right\} \tag{23}$$

$$\text{s.t.} \quad \sum_{p=0}^{n_s} \beta_{s,p} \sqrt{\sum_{i \in I_k} \sigma_i^2 \exp\left\{ \sum_{j=1}^{M} (2a_{ij} r_j + 2p x_k) \right\}}$$

$$+ \sum_{i \in I_k} E_{c_i} \exp\left\{ \sum_{j=1}^{M} a_{ij} r_j \right\} \leq 1, \ s = 1, \cdots, S, \ k = 1, \cdots, K, \tag{24}$$

$$\sum_{k=1}^{K} x_k \geq \log(1-\epsilon), \ x_k \leq 0, \ k = 1, \cdots, K. \tag{25}$$

As $E_{c_i} \geq 0$ and $\beta_{s,p} \geq 0$, problem (23)-(25) is a convex programming problem. Hence, interior point methods can be efficiently used to solve it and to provide a lower bound for the joint probabilistic constrained problem (4)-(5).

2.4. Bonferroni approximation

In order to come up with an upper bound of the joint probabilistic constrained problem (4)-(5), we adopt the popular Bonferroni approximation, which gives probabilistic measures to individual constraints. We give an estimation y_k^0 for each y_k, $k = 1, \cdots, K$, such that $\prod_{k=1}^{K} y_k^0 = 1 - \epsilon$ and $y_k^0 \geq 0$. We then use the following individual probabilistic constraint

$$P(\sum_{i \in I_k} c_i \prod_{j=1}^{M} t_j^{a_{ij}} \leq 1) \geq y_k^0, \ k = 1, \cdots, K. \tag{26}$$

to approximate the joint probabilistic constraint (5), and we have

Theorem 2. *With the estimation y_k^0, we have the following approximation of the stochastic geometric program with a joint probabilistic constraint (4)-(5):*

$$\min_t \quad \sum_{i \in I_0} E_{c_i} \prod_{j=1}^{M} t_j^{a_{ij}} \tag{27}$$

$$\text{s.t.} \quad \sum_{i \in I_k} E_{c_i} \prod_{j=1}^{M} t_j^{a_{ij}} + \Phi^{-1}(y_k^0) \sqrt{\sum_{i \in I_k} \sigma_i^2 \prod_{j=1}^{M} t_j^{2a_{ij}}} \leq 1, k = 1, \cdots, K, \tag{28}$$

The optimal value of the approximation problem (27)-(28) is an upper bound of problem (4)-(5).

Proof. We know from [3] that, when the parameters $c_i, i \in I_k, \forall K$ are normally distributed and independent of each other, the probabilistic constraints (26) are equivalent to the constraints (28). Moreover, the chosen y_k^0 is one possible allocation of the total tolerance probability, hence the optimal solution of problem (27)-(28) is a feasible solution for the original problem (4)-(5). Hence, it provides an upper bound. □

As $\Phi^{-1}(y_k^0) \geq 0$, problem (27)-(28) is a geometric program, hence it can be transformed into a convex programming problem and solved by interior point methods.

In order to improve the performance of Bonferroni approximation, we randomly generate several groups of the sample y_k^0, $k = 1, \cdots, K$ such that $\prod_{k=1}^{K} y_k^0 = 1 - \epsilon$ and $y_k \geq 0$. We solve the problem (27)-(28) with each group of y_k^0, $k = 1, \cdots, K$, and find its optimal value. Then we compare the obtained optimal values and find the minimal one. With larger sample size, we can find a better upper bound than Bonferroni approximation.

3. Numerical experiments

We consider a shape optimization problem with a joint probabilistic constraint as an example to test the performances of our approximations

$$\min_{h,w,d} \quad h^{-1}w^{-1}d^{-1} \tag{29}$$

$$\text{s.t.} \quad P\big((2/A_{wall})hw + (2/A_{wall})hd \le 1,\ (1/A_{flr})wd \le 1\big) \ge 1-\epsilon, \tag{30}$$

$$\alpha h^{-1}w \le 1,\ (1/\beta)hw^{-1} \le 1, \tag{31}$$

$$\gamma wd^{-1} \le 1,\ (1/\delta)w^{-1}d \le 1. \tag{32}$$

We set $\alpha = \gamma = 0.5$, $\beta = \delta = 2$, $\epsilon = 5\%$, and assume $1/A_{wall} \sim N(0.005, 0.01)$ and $1/A_{flr} \sim N(0.01, 0.01)$.

We use the polynomial approximation (20)-(22) and Bonferroni approximation (27)-(28) to estimate the lower bound and the upper bound for problem (29)-(32), respectively.

We carry out nine polynomial approximations, by setting the degrees of the polynomial functions to be from one to nine. For each of the polynomial functions, we use the problem (17)-(19) to estimate the coefficients. Then we carry out Bonferroni approximation with 1 group, 5 groups, 10 groups, 20 groups and 100 groups of y_1^0 and y_2^0. For each group, we randomly generate y_1^0 by an uniform distribution on $(1-\epsilon, 1)$, and compute $y_2^0 = (1-\epsilon)/y_1^0$. By using Sedumi solver[2] from CVX package [5], we solve the approximation problems with Matlab R2010b, on a PC with a 3.0 Ghz Intel Core2 Duo CPU and 4.0 GB RAM. Table 1 (Table 2) shows the degrees of the polynomials (the sample sizes), the variable numbers, the constraint numbers, the optimal values and the CPU time of the polynomial (Bonferroni) approximation problems. For better illustration, we compute the gaps of polynomial approximation bounds, which are the percentage differences between the bounds and the best Bonferroni approximation bound, and show them in the sixth column of Table 1. Correspondingly, Table 2 shows the gaps of Bonferroni approximation bounds, which are the percentage differences between them and the best polynomial approximation bound.

[2]The Variable Number account both the original variables and the slater variables, which are added to the programming to match the solver data input requirements. The Constraint Number account all the original constraints, the added constraints and the exponential constraints.

Table 1: Computational results of polynomial approximation with $\epsilon = 5\%$

Num.	Degree	Var. Num.*	Con. Num.**	Opt. Val.	Gap(%)	Cpu time(s)
1	1	175	87	0.211	17.606	1.420
2	2	265	138	0.230	10.324	2.195
3	3	376	204	0.233	9.098	2.163
4	4	490	273	0.233	8.990	2.434
5	5	601	339	0.234	8.879	2.645
6	6	714	407	0.234	8.766	2.553
7	7	825	473	0.234	8.647	2.756
8	8	936	539	0.235	8.499	3.684

Table 2: Computational results of Bonferroni approximation with $\epsilon = 5\%$

Num.	Sample size	Var. Num.	Con. Num.	Opt. Val.	Gap(%)	Cpu time(s)
1	1	129	69	0.271	15.699	2.126
2	5	129	69	0.257	9.611	7.456
3	10	129	69	0.256	9.358	14.168
4	20	129	69	0.256	9.342	25.273
5	100	129	69	0.256	9.288	127.389

From Table 1, we can see that as the degree of the polynomial function increases, the gap of the corresponding polynomial approximation bound becomes smaller. Meanwhile, although the problem size is obviously increasing with the degree of the polynomial function, the CPU time is not increasing sharply. This illustrates that the polynomial approximation problem is not hard to be solved by the interior point method. However, when the degree of the polynomial function is very large (larger than 8), the solver meets some very large numbers during the solution process. Therefore, it can not find an optimal solution. Due to this restriction on the degree, the gap of the polynomial approximation bound can not decrease to zero.

On the other hand, we can see from Table 2 that as the sample size increases, the gap of the corresponding Bonferroni approximation bound becomes smaller. This means that, adopting more samples can efficiently improve the performance of Bonferroni approximation method.

Although the size of Bonferroni approximation problem is fixed, we need to heuristically solve the problem many times to find a better y_k^0, $k = 1, \cdots, K$.

Moreover, we set $\epsilon = 2\%$ and $\epsilon = 1\%$, and test the polynomial approximation method and the Bonferroni approximation method for the two cases, respectively. The other parameters and the used methods are the same as the case with $\epsilon = 5\%$. The corresponding results are given in Table 3 and Table 4.

Table 3: Computational results of polynomial approximation with $\epsilon = 2\%$ and $\epsilon = 1\%$

		2%			1%		
Num.	Degree	Obj.	Gap(%)	Cpu time(s)	Opt. Val.	Gap(%)	Cpu time(s)
1	1	0.249	17.794	1.705	0.274	18.090	1.400
2	2	0.275	9.152	1.486	0.307	8.329	2.141
3	3	0.277	8.524	1.838	0.308	8.004	2.076
4	4	0.279	7.884	1.879	0.309	7.677	2.193
5	5	0.280	7.285	2.684	0.310	7.348	2.142
6	6	0.280	7.254	2.704	0.312	7.016	2.435
7	7	0.290	4.268	3.016	0.319	4.730	2.809
8	8	0.290	4.268	3.487		NaN	

Table 4: Computational results of Bonferroni approximation with $\epsilon = 2\%$ and $\epsilon = 1\%$

		2%			1%		
Num.	times	Obj.	Gap(%)	Cpu time(s)	Opt. Val.	Gap(%)	Cpu time(s)
1	1	0.313	7.978	1.597	0.364	14.114	1.925
2	5	0.304	4.991	6.290	0.336	5.187	6.376
3	10	0.302	4.464	11.525	0.336	5.187	13.592
4	20	0.302	4.458	22.127	0.335	4.976	26.336
5	100	0.302	4.458	107.023	0.335	4.965	118.748

From Table 3 and Table 4, we can see that the observations from the two former tables are still valid. Moreover, as the tolerance probability ϵ becomes smaller, the gaps between the polynomial approximation bound and the Bonferroni approximation bounds are smaller than when larger values of ϵ are considered.

Acknowledgement

This research was supported by the Programme Cai Yuanpei. The first author was financially supported by China Scholarship Council. The first and third author were supported by National Natural Science Foundation of China under Grant Numbers 71371152 and 11571270.

References

[1] S. Boyd, S. J. Kim, L. Vandenberghe, A. Hassibi, *A tutorial on geometric programming*, Optim. Eng. 8 (2007), 67–127.

[2] J. Cheng, A. Lisser, *A second-order cone programming approach for linear programs with joint probabilistic constraints*, Oper. Res. Letters 40 (2012), 325–328.

[3] J. Dupačová, *Stochastic geometric programming: approaches and applications*, In V. Brožová, R. Kvasnička, eds., Proceedings of MME09 (2009), 63–66.

[4] J. Gorski, F. Pfeuffer, K. Klamroth, *Biconvex sets and optimization with biconvex functions: a survey and extensions*, Mathematical Methods of Operations Research 66 (3) (2007), 373-407.

[5] M. Grant, S. Boyd, Y. Ye, *CVX: Matlab software for disciplined convex programming* (2008).

[6] K. Iwata, Y. Murotsu, T. Iwatsubo, *A probabilistic approach to the determination of the optimal cutting conditions*, J. Eng. Ind. Trans. ASME 4 (1972), 1099–1107.

[7] S.J. Kim, S. P. Boyd, S. Yun, D.D. Patil, M.A. Horowitz, *A heuristic for optimization stochastic activity networks with applications to statistical digital circuit sizing*, Optim. Eng. 8 (2007), 397–430

[8] M. Luptáčik, *Geometric programming: Method and applications*, OR Spektrum 2 (1981), 129–143

[9] L.B. Miller, H. Wagner, *Chance-constrained programming with joint constraints*, Oper. Res. 13 (1965) 930–945.

[10] A. Prékopa, *Stochastic Programming*, Kluwer Academic Publishers, Dordrecht, Boston, (1995).

[11] S.S. Rao, *Engineering Optimization: Theory and Practice*, 3rd Ed., Wiley-Interscience, New York, (1996).

[12] R.D., Wiebking, *Optimal engineering design under uncertainty by geometric programming*, Manag. Sci. 6 (1977), 644–651