Global Optimality Conditions and Optimization Methods for Constrained Polynomial Programming Problems[☆]

Zhiyou Wu[a,b,*], Jing Tian[b], Julien Ugon[b], Liang Zhang[a]

[a]*School of Mathematics, Chongqing Normal University, Chongqing 401331, P.R. China*
[b] *School of Science, Information Technology and Engineering, Federation University Australia, Ballarat 3353, Victoria, Australia*

Abstract

The general constrained polynomial programming problem (*GPP*) is considered in this paper. Problem (*GPP*) has a broad range of applications and is proved to be NP-hard. Necessary global optimality conditions for problem (*GPP*) are established. Then, a new local optimization method for this problem is proposed by exploiting these necessary global optimality conditions. A global optimization method is proposed for this problem by combining this local optimization method together with an auxiliary function. Some numerical examples are also given to illustrate that these approaches are very efficient.

Keywords: constrained polynomial programming problem, necessary global optimality condition, linear transformation, local optimization method, global optimization method.

1. Introduction

The general constrained polynomial programming problem (*GPP*) is widespread in the mathematical modeling of real world systems for a very broad range of applications. Such applications include engineering design, signal processing, speech recognition, material science, investment science, quantum mechanics, allocation and location problems, quadratic assignment and numerical linear algebra [1, 2]. Since polynomial functions are non-convex, the problem (*GPP*) is NP-hard, even when the objective function is quadratic and the feasible set is a simplex [3].

A classic approach for the problem (*GPP*) is convex relaxation methods [3, 4, 5]. Among various convex relaxation methods, semidefinite programming (SDP) and sum of squares (SOS) relaxations are very popular. Specifically, by representing each nonnegative polynomial as a sum of squares of some other polynomials, it is possible to relax each polynomial inequality as a convex linear matrix inequality(LMI) [6]. Theoretically, for Lasserre's SDP relaxation method, it was proved that when the feasible region of (*GPP*) is compact, its optimal value can be approximated within any accuracy by the sequence of SDP relaxations [7]. However, the practical solvability of SDP or SOS relaxation method depends on the size or the degree of the polynomial programming problem. Indeed, so far the most effective use of SDP relaxation has been for the quadratic optimization problems[6]. From a computational point of view, by SDP or SOS relaxation method, it is hard to achieve a close approximation to the optimal value of (*GPP*) without efficient methods for handling large scale semidefinite programs [7]. However, as the authors in [3] mentioned, so far there are few efficient numerical methods for solving large scale polynomial optimization problems. So, solving large scale SDPs still remains a computational challenge.

Recently, some researchers applied SDP relaxation methods to some special models. [8] provided approximation methods for complex polynomial optimization. In [8], the objective function takes three forms: multilinear, homogenous polynomial and a conjugate symmeric form. The constraint belongs to three sets: the m-th roots of complex unity, the complex unity and the Euclidean sphere. [9] established some approximation solution methods to

---

solve quadratically constrained multivariate bi-quadratic optimization. [6] presented a general semidefinite relaxation scheme for genear n-variate quartic polynomial optimization under homogeneous quadratic constraints. [2] considered approximation algorithms for optimizing a generic multi-variate homogeneous polynomial function, subject to homogenous quadratic constraints.

Global optimality conditions are very important in global optimization field. References [10]-[13] focus on global optimality conditions for the problems with quadratic objective function subject to linear constraints or quadratic constraints. Based on the so-called Positivstellensatz (a polynomial analogue of the transposition theorem for linear systems), it is possible to formulate global necessary and sufficient conditions for problems (GPP)[14]. [15] proved in Theorem 4.2 a sufficient conditions for global optimality in (GPP), which is a special case of global necessary and sufficient conditions in [14]. [16] provided another necessary and sufficient global optimality conditions for (GPP). However all these conditions are complex and difficult to check in practice since the conditions involve solving a sequence of semidefinite programs. As it mentioned in [14], only under the idealized assumptions that all semidefinite programs can be solved exactly, it is possible for these conditions to be checked.

In this paper, we consider the following general constrained polynomial programming problem (*GPP*).

$$
\begin{aligned}
(GPP) \quad \min \quad & f(x) \\
s.t. \quad & g_t(x) \leq 0, \ t = 1, \cdots, m \\
& x \in X,
\end{aligned}
$$

where $f : X \to R$, $g_t : X \to R$, $t = 1, \cdots, m$, and X is a box with $x_i \in [u_i, v_i], i = 1, \ldots, n$. $S = \{x \in X | g_t(x) \leq 0, \ t = 1, \cdots, m\}$ is feasible set.

In this paper, we will discuss necessary global optimality conditions for problem (*GPP*). These conditions are obtained by studying Karush-Kuhn-Tucker (KKT) conditions and a necessary and sufficient condition for a point being a global minimizer for a constrained univariate polynomial programming problem. Then a new strongly local optimization method will be designed for problem (GPP) according to the necessary global optimality conditions. The new strongly local optimization method improves traditional local optimization method which is based on KKT conditions. Finally, we will design a global optimization method to solve the problem (GPP) by combining the new strongly local optimization method and an auxiliary function. Numerical examples illustrate the efficiency of the optimization methods proposed in the paper.

The layout of the paper is as follows. Necessary global optimality conditions for the problem (*GPP*) are provided in section 2. A new strongly local optimization method and a global optimization method for the problem (*GPP*) are designed in section 3. Some numerical examples for the problem (*GPP*) are illustrated in section 4. Conclusion is presented in section 5.

2. Necessary Global Optimality Conditions for Problem (*GPP*)

In this section, we will provide necessary global optimality conditions to problem (*GPP*). Actually, we construct a point set where the global minimizer lies in. We can obtain the global minimizer by comparing the function values of all points in the set.

Firstly, we consider the following univariate polynomial optimization.

$$
\begin{aligned}
(UPP) \quad \min \quad & p(x) \\
s.t. \quad & q_t(x) \leq 0, \ t = 1, \cdots, m \\
& x \in [u, v].
\end{aligned}
$$

Let $\Omega = \{x \in [u, v] | q_t(x) \leq 0, \ t = 1, \cdots, m\}$.

The problem (*UPP*) is interesting not only because of the inherent simplicity of the problem strture and rich modeling capabilities, but also because this problem forms the backbone of multi-variate polynomial optimization [17].

2

For methods to solving the problem (UPP), please refer to [17]-[18] and the papers therein. [17] applies the global optimization algorithm (GOP) which proposed for solving constrained nonconvex problems involving quadratic and polynomial functions in the objective function and/or constraints in [19] to the special case of polynomial functions of one variable. It illustrates the effectiveness of the algorithm. [18] presents a significant enhancement of reformulation-linearization technique (RLT) and shows empirically that this approach yield very tight lower bounds.

Since the feasible set Ω is a compact set and is not easy to work out, we will construct a new point set $\Omega^0 \subset \Omega$. Let $\Omega^1 = \{u, v | q_t(u) \leq 0, q_t(v) \leq 0, \ t = 1, \cdots, m\}$, $\Omega^2 = \{x | \nabla p(x) = 0, q_t(x) < 0, \ t = 1, \cdots, m, x \in (u, v)\}$ and $\Omega_t^3 = \{x | q_t(x) = 0, q_j(x) \leq 0, \ j \neq t, \ j = 1, \cdots, m, x \in (u, v)\}$, $t = 1, \cdots, m$. Let

$$\Omega^0 = \Omega^1 \bigcup \Omega^2 \bigcup_{t=1}^{m} \Omega_t^3. \tag{1}$$

Remark 1. *Since $p(x)$ and $q_t(x)$, $t = 1, \cdots, m$, are univariate polynomials, we suppose that the degree of $p(x)$ is dp and the degrees of $q_t(x)$, are dq_t, $t = 1, \cdots, m$, respectively. We use following methods to work out these point sets Ω^1, Ω^2 and Ω_t^3, $t = 1, \cdots, m$:*

1. *u and v will be kept if $q_t(u) \leq 0$, $q_t(v) \leq 0$, $t = 1, \cdots, m$. So, $|\Omega^1| \leq 2$;*
2. *Calculate all stationary points of $p(x)$ in the interval (u, v) ($\{x \in (u, v) | \nabla p(x) = 0\}$) which will be kept if $q_t(x) < 0$, for all $t = 1, \cdots, m$. So, $|\Omega^2| \leq dp - 1$;*
3. *Calculate all roots of $q_t(x)$ in the interval (u, v) ($\{x \in (u, v) | q_t(x) = 0\}$), $t = 1, \cdots, m$, which will be kept if $q_j(x) \leq 0$, $j \neq t$, $j = 1, \cdots, m$. So, $|\Omega^3| \leq \sum_{t=1}^{m} dq_t$.*

When it comes to finding roots of a univariate polynomial, we refer to the methods in [20] and [21]. In our implementation, we use command 'roots' in Matlab to calculate all roots.

Proposition 1. *For the problem (UPP), let $\bar{x} \in \Omega$. $\bar{x}$ is a global minimizer of (UPP) over Ω if and only if the following condition holds:*

$$p(\bar{x}) \leq p(x), \quad \forall x \in \Omega^0, \tag{2}$$

where Ω^0 is defined in (1).

Proof. $\Rightarrow$ The proof is obvious since $\Omega^0 \subset \Omega$.
$\Leftarrow$ We suppose that $\bar{x}$ is not a global minimizer of $p(x)$ over Ω and x^* is a global minimizer of $p(x)$ over Ω. So we have $p(x^*) < p(\bar{x})$.

From the condition (2), we know that $x^* \in \Omega \setminus \Omega^0$ (which means $x^* \in \Omega$ and $x^* \notin \Omega^0$). By $x^* \notin \Omega^1$, we have $x^* \in (u, v)$. By $x^* \notin \bigcup_{t=1}^{m} \Omega_t^3$, we have $q_t(x^*) < 0$, $t = 1, \cdots, m$. By $x^* \notin \Omega^2$, $x^* \in (u, v)$ and $q_t(x^*) < 0$, $t = 1, \cdots, m$, we have $\nabla p(x^*) \neq 0$.

So, we have the following properties. Let $d = -\nabla p(x^*)$. There exists an $s > 0$, such that

1. $x^* + sd \in (u, v)$;
2. $q_t(x^* + sd) < 0$, for all $t = 1, \cdots, m$;
3. $p(x^* + sd) < p(x^*)$

So we can conclude $x^* + sd \in \Omega$ and $p(x^* + sd) < p(x^*)$, which contradicts that x^* is a global minimizer of $p(x)$ over Ω. ∎

By using Proposition 1, we will give necessary global optimality conditions for Problem (GPP). Let $\bar{x} \in S$, Q be an invertible matrix, let

$$x := Qy, \quad F(y) := f(Qy) = f(x), \quad \bar{y} := Q^{-1}\bar{x},$$

and let $(Q)_i$ represent the ith row of Q, $(Q)_{ij}$ represent the entry of Q in the ith row and the jth column.

Let $Y = \{y = Q^{-1}x | x \in X\}$. For $\bar{y} = (\bar{y}_1, \ldots, \bar{y}_n)^T = Q^{-1}\bar{x}$, let $y = (\bar{y}_1, \cdots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \cdots, \bar{y}_n)^T$. Let $\triangle_k = \sum_{\substack{j=1 \\ j \neq i}}^{n} (Q)_{kj}\bar{y}_j = \bar{x}_k - (Q)_{ki}\bar{y}_i = \bar{x}_k - (Q)_{ki}(Q^{-1})_i\bar{x}$, $k = 1, \cdots, n$, and let

$$
\begin{aligned}
l_i &= \max\left\{\min\left\{\frac{u_1 - \triangle_1}{(Q)_{1i}}, \frac{v_1 - \triangle_1}{(Q)_{1i}}\right\}, \cdots, \min\left\{\frac{u_n - \triangle_n}{(Q)_{ni}}, \frac{v_n - \triangle_n}{(Q)_{ni}}\right\}\right\}, \\
r_i &= \min\left\{\max\left\{\frac{u_1 - \triangle_1}{(Q)_{1i}}, \frac{v_1 - \triangle_1}{(Q)_{1i}}\right\}, \cdots, \max\left\{\frac{u_n - \triangle_n}{(Q)_{ni}}, \frac{v_n - \triangle_n}{(Q)_{ni}}\right\}\right\}.
\end{aligned}
$$

Then we can obtain the following results:

$$
\begin{aligned}
&(1) \quad l_i \leq r_i, \\
&(2) \quad [l_i, r_i] = \{y_i \mid (\bar{y}_1, \cdots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \cdots, \bar{y}_n)^T \in Y\}.
\end{aligned}
$$

Let $G_t(y_i) = g_t(Qy) = g_t(x)$. We have $S_i^1 = \{l_i, r_i | G_t(l_i) \leq 0, G_t(r_i) \leq 0 | t = 1, \cdots, m\}$, $S_i^2 = \{y_i | \nabla f(Qy) = 0, g_t(Qy) < 0, \ t = 1, \cdots, m, y = (\bar{y}_1, \cdots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \cdots, \bar{y}_n)^T, y_i \in (l_i, r_i)\}$ and $S_{t,i}^3 = \{y_i | g_t(Qy) = 0, g_j(Qy) \leq 0, \ j \neq t, \ j = 1, \cdots, m, y = (\bar{y}_1, \cdots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \cdots, \bar{y}_n)^T, y_i \in (l_i, r_i)\}$, $t = 1, \cdots, m$, $i = 1, \cdots, n$. Let

$$
S_i^0 = S_i^1 \bigcup S_i^2 \bigcup_{t=1}^{m} S_{t,i}^3. \tag{3}
$$

Let us review Karush-Kuhn-Tucker (KKT) conditions for problem (GPP).

If $\bar{x}$ is a local optimal solution, then the following KKT conditions hold under some constraint qualifications: there exist nonnegative scalars α_t, $t = 1, \cdots, m$, β_i and γ_i, $i = 1, \cdots, n$, such that

$$
[KKT] \begin{cases} \nabla f(\bar{x}) + \sum_{t=1}^{m} \alpha_t \nabla g_t(\bar{x}) + \beta - \gamma = 0, \\ \alpha_t g_t(\bar{x}) = 0, \ t = 1, \cdots, m \\ \beta(x - v) = 0 \\ \gamma(u - x) = 0 \end{cases},
$$

where $\beta = (\beta_1, \cdots, \beta_n)^T$ and $\gamma = (\gamma_1, \cdots, \gamma_n)^T$. See [22] for various constraint qualifications, such as Abadie constraint qualification, linearity constraint qualification, Slater's constraint qualification, linear independence constraint qualification, Cottle's constraint qualification, Zangwill's constraint qualification, Kuhn-Tucker's constraint qualification.

Theorem 1. *(Necessary global optimality conditions for (GPP)) Let $\bar{x} \in S$ and Q be any invertible matrix. If $\bar{x}$ is a global minimizer of (GPP), then*

$$
[GNC] \begin{cases} [KKT] \text{ conditions hold under some constraint qualifications;} \\ [NC]_i : \ f(\bar{x}) \leq f(x), \ \forall (Q^{-1})_i x \in S_i^0, \ \forall i = 1, \cdots, n \end{cases}
$$

where S_i^0 is defined in (3).

Proof. If $\bar{x}$ is a global minimizer of (GPP), then it is also a local minimizer of (GPP). So under some constraint qualifications, KKT conditions hold.

Next, we prove conditions $[NC]_i$ hold. If $\bar{x}$ is a global minimizer of (GPP), then $f(\bar{x}) \leq f(x)$, for any $x \in S$.

Let $\bar{y} = Q\bar{x}$. For any $y = (\bar{y}_1, \ldots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \ldots, \bar{y}_n)^T \in Y$, i.e., $y_i \in [l_i, r_i]$, $i = 1, \ldots, n$, let $x = Qy$. Then $x \in X$. So we have $f(Q\bar{y}) \leq f(Qy)$, for any $y_i \in [l_i, r_i]$, $i = 1, \ldots, n$. By using Proposition 1, we have the following conditions $[NC]_i$ hold:

$$
[NC]_i \quad f(\bar{x}) \leq f(x), \ \forall (Q^{-1})_i x \in S_i^0, \ \forall i = 1, \cdots, n.
$$

∎

4

Remark 2. *From Theorem [1], we can see that the global optimality conditions [GNC] are stronger than KKT conditions, since [GNC] include KKT conditions.*

Next, we take Problem 8 in section [4] for example to show [KKT] $\nsupseteq$ [NC]$_i$, for every $i = 1, \cdots, n$ below.

We fix $Q = I$ and choose two points $\bar{x} = (2.3295, 3.1785)^T$ which is a global minimizer and $\bar{y} = (1.5996, 2.8204)^T$ which is a local minimizer. It is easy to check that both [NC]$_i$ and [KKT] hold at $\bar{x}$, while [KKT] holds at $\bar{y}$, but [NC]$_i$ does not hold at $\bar{y}$.

In fact, $\bar{x} \in int(X)$, $\nabla f(\bar{x}) = (-1, -1)^T$ and $g_1(\bar{x}) = g_2(\bar{x}) = 0$, which means $\bar{x} \in S_{t,i}^3 \subset S_i^0$, $t = 1, 2$, $i = 1, 2$.

When $i = 1$ and we fix $\bar{x}_2 = 3.1785$, we have $S_1^1 = \emptyset$, $S_1^2 = \emptyset$, $S_{1,1}^3 = \{2.3295, 0.5179\}$ and $S_{2,1}^3 = \{2.3295, 0.6247\}$. But $f((0.5179, 3.1785)^T) = -3.6964 > f(\bar{x}) = -5.5080$ and $f((0.6247, 3.1785)^T) = -3.8033 > f(\bar{x}) = -5.5080$. So $f(\bar{x}) \leq f(x)$, for any $x \in S_1^0 = S_1^1 \bigcup S_1^2 \overset{2}{\underset{t=1}{\bigcup}} S_{t,1}^3$.

When $i = 2$ and we fix $\bar{x}_1 = 2.3295$, we have $S_2^1 = \{0\}$, $S_2^2 = \emptyset$, $S_{1,2}^3 = \{3.1785\}$ and $S_{2,2}^3 = \{3.1785\}$. But $f((2.3295, 0)^T) = -2.3295 > f(\bar{x}) = -5.5080$. So $f(\bar{x}) \leq f(x)$, for any $x \in S_2^0 = S_2^1 \bigcup S_2^2 \overset{2}{\underset{t=1}{\bigcup}} S_{t,2}^3$.

This means conditions [NC]$_i$, $i = 1, 2$, hold at $\bar{x}$.

Since $\nabla g_1(\bar{x}) = (-8.1639, 1)^T$ and $\nabla g_2(\bar{x}) = (4.6996, 1)^T$, we can find nonnegative scalars $\alpha_1 = 0.2876$ and $\alpha_2 = 0.7124$ such that [KKT] holds at $\bar{x}$.

While $\bar{y} \in int(X)$, $\nabla f(\bar{y}) = (-1, -1)^T$ and $g_1(\bar{y}) = g_2(\bar{y}) = 0$, which means $\bar{y} \in S_{t,i}^3 \subset S_i^0$, $t = 1, 2$, $i = 1, 2$.

When $i = 1$ and we fix $\bar{y}_2 = 2.8204$, we have $S_1^1 = \emptyset$, $S_1^2 = \emptyset$, $S_{1,1}^3 = \{2.2808, 1.5996, 0.4004\}$. $f((2.2808, 2.8204)^T) = -5.1012$, $f((1.5996, 2.8204)^T) = -4.4200$ and $f((0.4004, 2.8204)^T) = -3.2208$. So $f(\bar{y}) \leq f(x)$, for any $x \in S_1^0 = S_1^1 \bigcup S_1^2 \overset{2}{\underset{t=1}{\bigcup}} S_{t,1}^3$ does not hold at $\bar{y}$. This means [NC]$_1$ does not hold at $\bar{y}$.

Since $\nabla g_1(\bar{y}) = (3.0723, 1)^T$ and $\nabla g_2(\bar{x}) = (-5.3793, 1)^T$, we can find nonnegative scalars $\alpha_1 = 0.7548$ and $\alpha_2 = 0.2452$ such that [KKT] holds at $\bar{y}$.

3. Optimization methods for (*GPP*)

3.1. A New Local Optimization Method for Problem (*GPP*)

Definition 1. Let $\bar{x} \in S$ and Q be an invertible matrix. $\bar{x}$ is said to be a strongly local minimizer of problem (*GPP*) with respect to Q iff $\bar{x}$ satisfies the necessary global optimality conditions [*GNC*].

Definition 2. Let $\bar{x} \in S$ and Q be an invertible matrix. $\bar{x}$ is said to be a $\varepsilon-$strongly local minimizer of problem (*GPP*) with respect to Q iff KKT conditions hold at $\bar{x}$ and for any $i = 1, \cdots, n$, either $\bar{x}$ satisfies the condition [*NC*]$_i$ or there exists a point $X_i^* \in S$, such that X_i^* satisfies the condition [*NC*]$_i$ when $\bar{x}$ is replaced by X_i^*, and $|f(\bar{x}) - f(X_i^*)| \leq \varepsilon$.

Algorithm 1. *Strongly or $\varepsilon-$strongly local optimization method for problem (GPP):(S LOM).*
Step 0. Take an initial point $x_0 \in S$. Let $Q_1 = I$, $Q_2, \cdots, Q_s, \cdots, Q_N$ be any invertible matrices given randomly, where I is the identity matrix. Let ε be a small positive number. Let $s := 1$ and $Q := Q_s$ and $i = 1$. Let $x^* := (x_1^*, \cdots, x_n^*)^T$ be a local minimizer or *KKT* point of $f(x)$ on feasible set S starting from x_0. Let $\bar{x} := x^*$ and go to step 1;
Step 1. Let $\bar{y} = Q^{-1}\bar{x} = (\bar{y}_1, \ldots, \bar{y}_i, \ldots, \bar{y}_n)^T$, $y = (\bar{y}_1, \ldots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \ldots, \bar{y}_n)^T$ and $x = Qy$. Calculate S_i^1, and then check whether the condition holds:

$$f(\bar{x}) \leq f(Qy) + \varepsilon, \ \forall y = (\bar{y}_1, \ldots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \ldots, \bar{y}_n)^T \ and \ y_i \in S_i^1.$$

If this condition holds, go to step 2, otherwise set $\tilde{S} = S_i^1$ and go to step 4;
Step 2. Calculate S_i^2, and then check whether the condition holds:

$$f(\bar{x}) \leq f(Qy) + \varepsilon, \ \forall y = (\bar{y}_1, \ldots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \ldots, \bar{y}_n)^T \ and \ y_i \in S_i^2.$$

If this condition holds, go to step 3, otherwise set $\tilde{S} = S_i^2$ and go to step 4;

Step 3. Set $t = 1$. Calculate $S_{t,i}^3$, and then check whether the condition holds:

$$f(\bar{x}) \leq f(Qy) + \varepsilon, \ \forall y = (\bar{y}_1, \ldots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \ldots, \bar{y}_n)^T \ and \ y_i \in S_{t,i}^3.$$

If the condition holds, set $t = t + 1$ and repeat to check the condition until $t = m$ and go to step 5; otherwise set $\tilde{S} = S_{t,i}^3$ and go to step 4;

Step 4. Let $\bar{y}_i^* := argmin\{f(Qy)|y = (\bar{y}_1, \ldots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \ldots, \bar{y}_n)^T \ and \ y_i \in \tilde{S}\}$ and $\bar{y}^* = (\bar{y}_1, \cdots, \bar{y}_{i-1}, \bar{y}_i^*, \bar{y}_{i+1} \cdots, \bar{y}_n)^T$. Let $\bar{x}^* := Q\bar{y}^*$. Let $x^* = (x_1^*, \cdots, x_n^*)$ be a local minimizer or KKT point of $f(x)$ on S starting from $\bar{x}^*$. If $f(x^*) < f(\bar{x}) - \varepsilon$, let $\bar{x} := x^*$, $i := 1$, $s := 1$ and $Q := Q_s$, go to Step 1; otherwise go to Step 5;

Step 5. If $i := n$, go to Step 6; otherwise, let $i := i + 1$ and go to Step 1;

Step 6. Let $s := s + 1$. If $s > N$, go to Step 7; otherwise, let $Q := Q_s$ and $i := 1$, go to Step 1;

Step 7. Stop. $\bar{x}$ is a strongly or $\varepsilon-$strongly local minimizer with respect to Q_s, $s = 1, \cdots, N$.

Remark 3. *In step 0 and step 4, we can apply any local optimization algorithm to get local minimizer or KKT point, such as feasible direction methods, penalty function methods, starting from $\bar{x}$. In our implementation, the optimization subroutine 'fmincon' within the optimization Toolbox in Matlab is used as the local search scheme to obtain local minimizers.*
In step 1, step 2 and step 3, we need to calculate S_i^1, S_i^2 and $S_{t,i}^3$, $t = 1, \cdots, m$. For any $i \in \{1, \cdots, n\}$, let $\bar{x} \in S$, $\bar{y} = Q^{-1}\bar{x}$ and $y = (\bar{y}_1, \cdots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \cdots, \bar{y}_n)^T$, where $y_i \in [l_i, r_i]$. Then $f(Qy)$ and $g_t(Qy)$, $t = 1, \cdots, m$, are univariate polynomials. So, we refer to Remark 1 to calculate these point sets.

Theorem 2. *For a given initial point $x_0 \in S$, we can obtain a strongly or $\varepsilon-$strongly local minimizer $\bar{x}$ of problem (GPP) in finite iteration times by the given strongly local optimization method (SLOM).*

Proof: First, we can prove that this algorithm must stop in finite iteration times.
Let $M := max\{f(x)|x \in S\}$ and $m := min\{f(x)|x \in S\}$. For the given Q_s, there are at most $n\frac{M-m}{\varepsilon}$ iteration times from *step 1* to *step 5*. In fact, for the given Q_s and given i, if $[NC]_i$ holds or if $f(x^*) \geq f(\bar{x}) - \varepsilon$, then we will change the i into $i + 1$; only when $[NC]_i$ does not hold and $f(x^*) < f(\bar{x}) - \varepsilon$, we will change i to 1 in *step 4* and go to *step 1*. For the same Q_s, when we change i to 1, the objection function value will decrease at least ε. Hence, there are at most $\frac{M-m}{\varepsilon}$ times to change i to 1 in *step 4*. The total iteration times from *step 1* to *step 5* are at most $n\frac{M-m}{\varepsilon}$. Since we have N numbers of Q_s, this algorithm must stop at most $Nn\frac{M-m}{\varepsilon}$ iteration times.

Second, let L be the set of all the KKT points of problem (GPP), and let $L_f := \{f(x) \mid x \in L\}$. We can prove that

(1) If L_f is a finite set, then we can obtain a strongly local minimizer in finite iteration times when ε is a very small number. In fact, let $\eta := min\{|f(x) - f(y)| \mid x, y \in L \ and \ f(x) \neq f(y)\}$. Since L_f is a finite set, we have that $\eta > 0$. When $\varepsilon < \eta$, we know that $f(x^*) < f(\bar{x}) - \varepsilon$ in *step 4* is equivalent to $f(x^*) < f(\bar{x})$. Hence, for the given Q_s and given i, if $[NC]_i$ holds, then we will change the i into $i + 1$; if $[NC]_i$ does not hold in *step 1* or *step 2* or *step 3* which means that $f(\bar{x}) > min\{f(Qy)|y = (\bar{y}_1, \ldots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \ldots, \bar{y}_n)^T \ and \ y_i \in \tilde{S}\}$, then in *step 4*, we will find point $\bar{y}_i^*$ such that $f(Q\bar{y}^*) = min\{f(Qy)|y = (\bar{y}_1, \ldots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \ldots, \bar{y}_n)^T \ and \ y_i \in \tilde{S}\}$. Hence, we have that $f(x^*) < f(\bar{x})$ since $f(x^*) \leq f(Q\bar{y}^*) < f(\bar{x})$ and we have $x^* \in L$. Therefore, for the given Q_s and given i, if $[NC]_i$ does not hold in *step 1* or *step 2* or *step 3*, then we can obtain a new KKT point x^* such that $f(x^*) < f(\bar{x})$ which also satisfies that $f(x^*) < f(\bar{x}) - \varepsilon$. Hence, for the given Q_s, we can find a point $\bar{x}$ which satisfies all the condition $[NC]_i, i = 1, \ldots, n$ in at most $n\frac{M-m}{\varepsilon}$ iteration times. Therefore, in finite times, we can obtain a strongly local minimizer of problem (GPP) for all $Q_s, s = 1, \ldots, N$.

(2) If L_f is an infinite set, then we can obtain an $\varepsilon-$ strongly local minimizer in finite iteration times.
By the algorithm, for the given Q_s and given i, if $[NC]_i$ holds or if $f(x^*) \geq f(\bar{x}) - \varepsilon$, then we will change the i into $i + 1$; if $[NC]_i$ does not hold and $f(x^*) < f(\bar{x}) - \varepsilon$, then in *step 4*, we will find point $\bar{y}_i^*$ such that $f(Q\bar{y}^*) = min\{f(Qy)|y = (\bar{y}_1, \ldots, \bar{y}_{i-1}, y_i, \bar{y}_{i+1}, \ldots, \bar{y}_n)^T \ and \ y_i \in \tilde{S}\}$, where $\bar{y}_i^*$ satisfies condition $[NC]_i$. Since this algorithm must stop in finite steps, the final obtained point $\bar{x}$ must satisfy the following condition: for the given Q_s and given i, $[NC]_i$ holds or $f(Q\bar{y}^*) \geq f(x^*) \geq f(\bar{x}) - \varepsilon$, where $\bar{y}_i^*$ satisfies the condition $[NC]_i$. Hence $\bar{x}$ is an $\varepsilon-$ strongly local minimizer of problem (GPP). ∎

6

3.2. A global Optimization Method for Problem (GPP)

In this section, we will design a global optimization method for Problem (*GPP*) by combining the strongly local optimization method and an auxiliary function. In this paper, we use the following auxiliary function which is introduced in reference [23]. For the detailed properties, see [23].

For any $r > 0$ and given $c > 0$, let

$$g_{r,c}(t) = \begin{cases} c, & t \geq 0 \\ -\dfrac{2c}{r^3}t^3 - \dfrac{3c}{r^2}t^2 + c, & -r < t \leq 0 \\ 0, & t \leq -r \end{cases} \tag{4}$$

$$h_{r,c}(t) = \begin{cases} t + r, & t \leq -r \\ \dfrac{r-2}{r^3}t^3 + \dfrac{r-3}{r^2}t^2 + 1, & -r < t \leq 0 \\ 1, & 0 < t \leq 1 \\ -\dfrac{4c-2}{r^3}t^3 + \dfrac{(6c-3)(r+2)}{r^3}t^2 - \dfrac{(6c-3)(2r+2)}{r^3}t + \\ \quad \frac{4c-2+(6c-3)r}{r^3} + 1, & 1 < t \leq 1 + r \\ 2c, & t > 1 + r \end{cases} \tag{5}$$

Let

$$F_{q,r,c,x^*}(x) = q(exp(-\frac{\|x - x^*\|^2}{q})g_{r,c}\Big(f(x) - f(x^*)\Big) + h_{r,c}\Big(f(x) - f(x^*)\Big)), \tag{6}$$

where x^* is the current local minimum.

In the following, we will introduce a global optimization method to find a global minimizer of problem (*GPP*). The procedure of this global optimization method in the following consists of three phase circle:

Phase 1: (Strongly Local Search) Start from a given feasible point x_k and use strongly local minimization method *Algorithm 1* to search for a strongly local minimizer x_k^*.

Phase 2: (Local Search) Construct an auxiliary function $F_{q,r,c,x_k^*}(x)$. Find a KKT point or a local minimizer $\bar{x}_{q,r,c,x_k^*}$ of the function $F_{q,r,c,x_k^*}(x)$ over feasible set S.

Phase 3: (Global Search) If $\bar{x}_{q,r,c,x_k^*}$ is better than x_k^*, then let $k := k + 1$, $x_k := \bar{x}_{q,r,c,x_k^*}$ and return to Phase 1. Otherwise, stop the iteration process and return the incumbent local optimal solution x_k^* as a global optimal solution to the original problem.

Algorithm 2. *Global optimization method for problem (GPP):(GOM).*

Step 0. Set $M := 10^{10}$, $\mu := 10^{-10}$ and $k_0 := 2n$. Set $A_{n \times n} := I_{n \times n}$ and $B_{n \times 2n} := [A, -A]$. Let $r_0 := 1$, $c_0 := 1$, $q_0 := 10^5$ and $\delta_0 := \frac{1}{2}$. Let $k := 1$, $i := 1$ and $r := r_0$. Let x_1^0 be an initial point and $x_0^* := x_1^0$, then go to Step 1;

Step 1. Use the strongly or $\varepsilon-$strongly local optimization method (*SLOM*) to solve problem (*GPP*) starting from x_k^0. Let x_k^* be the obtained strongly or $\varepsilon-$strongly local minimizer of problem (*GPP*). If $f(x_k^*) \geq f(x_0^*)$, then go to step 6; otherwise let $q := q_0$, $c := c_0$, $r := r_0$, $\delta := \delta_0$, $i := 1$ and $x_0^* := x_k^*$, $k := k + 1$, then go to Step 2;

Step 2. Let B_i indicate the ith column of B and $\bar{x}_k^* := x_0^* + \delta B_i$. If $\bar{x}_k^* \notin S$, go to Step 3. Otherwise, if $f(\bar{x}_k^*) < f(x_0^*)$, then set $x_{k+1}^0 := \bar{x}_k^*$ and $x_0^* := \bar{x}_k^*$, $k := k + 1$ and go to Step 1; else go to Step 4;

Step 3. If $\delta < \mu$, go to Step 8; otherwise, let $\delta = \frac{\delta}{2}$ and go to Step 2;

Step 4. If $f(x_0^*) \leq f(\bar{x}_k^*) \leq f(x_0^*) + 1$, then go to Step 5; otherwise let $\delta = \frac{\delta}{2}$ go to Step 2;

Step 5. Let

$$F_{q,r,c,x_0^*}(x) = q(exp(-\frac{\|x - x_0^*\|^2}{q})g_{r,c}\Big(f(x) - f(x_0^*)\Big) + h_{r,c}\Big(f(x) - f(x_0^*)\Big)).$$

Solve the problem:

$$\begin{aligned} \min \quad & F_{q,r,c,x_0^*}(x) \\ s.t. \quad & x \in S. \end{aligned} \tag{7}$$

by a local search method starting from the initial point $\bar{x}_k^*$. Let $\bar{x}_{q,r,c,x_k^*}$ be the local minimizer obtained. Then set $x_{k+1}^0 := \bar{x}_{q,r,c,x_k^*}$, $k := k + 1$ and go to Step 1;

Step 6. If $q < M$, then increase q (in the following examples, let $q := 10q$), then go to Step 5; otherwise go to Step 7;

Step 7. If $c < M$, then increase c (in the following examples, let $c := 10c$), and let $q := q_0$, then go to Step 5; otherwise go to Step 8;

Step 8. If $i < k_0$, then let $i := i + 1$, $q := q_0$, $c := c_0$, $\delta = \delta_0$, go to Step 2; otherwise go to Step 9;

Step 9. If $r > \mu$, then decrease r (in the following examples, let $r := \frac{r}{10}$). Randomly select an orthogonal matrix $A_{n \times n}$ and set $B_{n \times 2n} := [A, -A]$. Let $i := 1$, $q := q_0$, $c := c_0$, $\delta = \delta_0$ and go to Step 2; otherwise, stop and x_0^* is the obtained global minimizer or approximate global minimizer of problem (GPP).

4. Numerical examples

In this section, we apply our two Algorithms: strongly local optimization method (SLOM) and global optimization method (GOM) to fifteen test problems. Table 1 shows summary information of the fifteen test problems. These test problems include Problems 1, 6-9 and 14 from the book [1], 10-12 form the paper [5] and 2-5, 13, 15 from the web site below:

$http : //www − optima.amp.i.kyoto − u.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page422.htm$.
For the detailed information of these problems, see the appendix in the end.

Table 1: Test problems for polynomial programming problems with polynomial constraints

Number of problems	Global minimizer x^*	Optimal value $f(x^*)$
5.1	$(0.5, 0, 3)$	-4
5.2	$(1, \cdots, 1, 3, 3, 3, 1)$	-15
5.3	$(2.171996, 2.363683, 8.773926, 5.095984, 0.9906548,$ $1.430574, 1.321644, 9.828726, 8.280092, 8.375927)$	24.3062091
5.4	$(14.095, 0.84296)$	-6961.81388
5.5	$(2.330499, 1.951372, -0.4775414, 4.365726,$ $-0.6244870, 1.038131, 1.594227)$	680.6300573
5.6	$(5, 1, 5, 0, 5, 10)$	-310
5.7	$(78, 33, 29.9953, 45, 36.7758)$	-30665.5387
5.8	$(2.3295, 3.1783)$	-5.5079
5.9	$(579.3167, 1359.943, 5110.071, 182.0174,$ $295.5985, 217.9799, 286.4162, 395.5979)$	7049.3307
5.10	†1	-575.5928
5.11	†2	-1.0178
5.12	†3	-153.6180
5.13	$\pm(1/2^{0.5}, 1/2)$	0.75
5.14	$(40.71751, 1.470)$	-16.73889
5.15	$(1/n^{0.5}, \cdots, 1/n^{0.5})$	-1

†1 $= -(0.4034, 0.4274, 0.4486, 0.4674, 0.4839, 0.4983, 0.5107, 0.5211, 0.5296, 0.5363, 0.5410, 0.5437, 0.5444,$ $0.5430, 0.5393)$;
†2 $= -(0.2418, 0.2208, 0.2085, 0.2000, 0.1934, 0.1882, 0.1838, 0.1800, 0.1767, 0.1738, 0.1712, 0.1688, 0.1667,$ $0.1647, 0.1629, 0.1612)$;
†3 $= -(-0.3642, 0.3955, 0.5042, 0.5589, 0.5892, 0.6049, 0.6109, 0.6104, 0.6057, 0.5991, 0.5828, 0.5173, 0.5193,$ $0.5306, 0.5459, 0.5619, 0.5763, 0.5869, 0.5919, 0.5896)$.

There are equalities involved in Problem 13-15. We can use our algorithms to solve them by converting equalities $h_s(x) = 0$, $s = 1, \cdots, l$ into equivalent inequalities $h_s(x) \leq 0$, $s = 1, \cdots, l$ and $-h_s(x) \leq 0$, $s = 1, \cdots, l$.

8

For our experiments, we use the optimality gap mentioned in [24] is:

$$GAP = |f(x) - f(x^*)|$$

where x is a heuristic solution obtained by our method and x^* is the optimal solution. We then say that a heuristic solution x is optimal if:

$$GAP \leq \begin{cases} \varepsilon & f(x^*) = 0 \\ \varepsilon \times |f(x^*)| & f(x^*) \neq 0 \end{cases}$$

In our experimentation we set $\varepsilon = 0.001$ as the same of that in [24].

In the table below, some common statistics are included. We randomly select 30 initial points for every problem. The *suc.rate*(success rate) means the success times out of 30. The *best* is the minimum of the results, the *worst* indicates the maximum of the results, and then it follows the *mean*, *median* and *st.dev.*(standard deviation). We also record the *av.it.*(average iteration times) and *av.CPU*(average CPU time) (in seconds). In some way, these statistics are able to evaluate the search ability and solution accuracy, reliability and convergence as well as stability.

Table 2: Results of algorithms SLOM and GOM for (GPP)

Problem	statistic	SLOM	GOM
1	suc.rate	30/30	30/30
	best	−4.0000	−4.0000
	worst	−4.0000	−4.0000
	mean	−4.0000	−4.0000
	median	−4.0000	−4.0000
	st.dev.	$2.7262e - 006$	$2.7262e - 006$
	av.it.	1	1
	av.CPU	0.1125	0.1125
2	suc.rate	30/30	30/30
	best	−15.0000	−15.0000
	worst	−15.0000	−15.0000
	mean	−15.0000	−15.0000
	median	−15.0000	−15.0000
	st.dev.	$8.9121e - 006$	$8.9121e - 006$
	av.it.	1.2333	1.2333
	av.CPU	0.2125	0.2125
3	suc.rate	30/30	30/30
	best	24.3062	24.3062
	worst	24.3062	24.3062
	mean	24.3062	24.3062
	median	24.3062	24.3062
	st.dev.	$4.9274e - 006$	$4.9274e - 006$
	av.it.	1	1
	av.CPU	0.1156	0.1156
4	suc.rate	30/30	30/30
	best	$−6.9618e + 003$	$−6.9618e + 003$
	worst	$−6.9618e + 003$	$−6.9618e + 003$
	mean	$−6.9618e + 003$	$−6.9618e + 003$
	median	$−6.9618e + 003$	$−6.9618e + 003$
	st.dev.	$8.0994e - 004$	$8.0994e - 004$
	av.it.	1	1
	av.CPU	0.0776	0.0776

continue goes here. . .

Problem	statistic	SLOM	GOM
5	suc.rate	30/30	30/30
	best	680.6301	680.6301
	worst	680.6301	680.6301
	mean	680.6301	680.6301
	median	680.6301	680.6301
	st.dev.	$5.3698e - 006$	$5.3698e - 006$
	av.it.	1	1
	av.CPU	0.1552	0.1552
6	suc.rate	26/30	30/30
	best	-310.0000	-310.0000
	worst	-184.0000	-310.0000
	mean	-293.2000	-310.0000
	median	-310.0000	-310.0000
	st.dev.	43.5640	$5.9702e - 006$
	av.it.	36.7667	37.8333
	av.CPU	5.5177	6.0469
7	suc.rate	30/30	30/30
	best	$-3.0666e + 004$	$-3.0666e + 004$
	worst	$-3.0666e + 004$	$-3.0666e + 004$
	mean	$-3.0666e + 004$	$-3.0666e + 004$
	median	$-3.0666e + 004$	$-3.0666e + 004$
	st.dev.	$4.4270e - 004$	$4.4270e - 004$
	av.it.	1	1
	av.CPU	0.0807	0.0807
8	suc.rate	30/30	30/30
	best	-5.5080	-5.5080
	worst	-5.5080	-5.5080
	mean	-5.5080	-5.5080
	median	-5.5080	-5.5080
	st.dev.	$9.9335e - 007$	$9.9335e - 007$
	av.it.	127.5333	127.5333
	av.CPU	7.7375	7.7375
9	suc.rate	29/30	30/30
	best	$7.0492e + 003$	$7.0492e + 003$
	worst	$8.7331e + 003$	$7.0492e + 003$
	mean	$7.1054e + 003$	$7.0492e + 003$
	median	$7.0492e + 003$	$7.0492e + 003$
	st.dev.	307.4294	$1.0895e - 006$
	av.it.	17.6000	17.6000
	av.CPU	4.3823	4.3823
10	suc.rate	30/30	30/30
	best	-575.5925	-575.5925
	worst	-575.5925	-575.5925
	mean	-575.5925	-575.5925
	median	-575.5925	-575.5925
	st.dev.	$1.9967e - 006$	$1.9967e - 006$
	av.it.	19	19
	av.CPU	83.0062	83.0062

continue goes here...

Problem	statistic	SLOM	GOM
11	suc.rate	7/30	30/30
	best	-1.1078	-1.1078
	worst	-0.0108	-1.1078
	mean	-0.2692	-1.1078
	median	-0.0144	-1.1078
	st.dev.	0.4706	$1.6607e - 014$
	av.it.	138.6000	143.4000
	av.CPU	$1.1422e + 003$	$1.6737e + 003$
12	suc.rate	30/30	30/30
	best	-153.6180	-153.6180
	worst	-153.6180	-153.6180
	mean	-153.6180	-153.6180
	median	-153.6180	-153.6180
	st.dev.	$7.5214e - 007$	$7.5214e - 007$
	av.it.	25	25
	av.CPU	43.1750	43.1750
13	suc.rate	30/30	30/30
	best	0.7500	0.7500
	worst	0.7500	0.7500
	mean	0.7500	0.7500
	median	0.7500	0.7500
	st.dev.	$6.2234e - 009$	$6.2234e - 009$
	av.it.	1	1
	av.CPU	0.0849	0.0849
14	suc.rate	30/30	30/30
	best	-16.7389	-16.7389
	worst	-16.7389	-16.7389
	mean	-16.7389	-16.7389
	median	-16.7389	-16.7389
	st.dev.	$5.8438e - 007$	$5.8438e - 007$
	av.it.	1	1
	av.CPU	0.0734	0.0734
15	suc.rate	30/30	30/30
	best	-1.0000	-1.0000
	worst	-1.0000	-1.0000
	mean	-1.0000	-1.0000
	median	-1.0000	-1.0000
	st.dev.	$8.2074e - 007$	$8.2074e - 007$
	av.it.	25	25
	av.CPU	8.1156	8.1156

It is shown from table 2 that GOM successfully solves all number of test problems and is very efficient and stable. As a local optimization method, SLOM can also be considered as a competitive algorithm with producing impressive results.

Next, we try to compare our GOM method with the other two methods: Interior Point Method and solver GloptiPoly 3. In our implementation, we execute the command 'fmincon' with algorithm of interior-point in Matlab. Solver GloptiPoly 3 is a Matlab/SeDuMi add-on for SDP-relaxations of minimization problems over multivariable polynomial functions subject to polynomial or integer constraints [25, 26].

Table 3: Comparisons between GOM, Interior-Point and GloptiPoly 3 for (GPP)

Problem	statistic	GOM	Interior Point	GloptiPoly 3
1	suc.rate	30/30	30/30	30/30
				order= 4
	best	−4.0000	−4.0000	−4.0000
	worst	−4.0000	−4.0000	−4.0000
2	suc.rate	30/30	29/30	30/30
				order= 2
	best	−15.0000	−15.0000	−15.0000
	worst	−15.0000	−12.6562	−15.0000
3	suc.rate	30/30	30/30	30/30
	best	24.3062	24.3062	24.3062
	worst	24.3062	24.3062	24.3062
4	suc.rate	30/30	30/30	30/30
	best	$-6.9618e + 003$	$-6.9618e + 003$	$-6.9618e + 003$
	worst	$-6.9618e + 003$	$-6.9618e + 003$	$-6.9618e + 003$
5	suc.rate	30/30	30/30	30/30
				order= 3
	best	680.6301	680.6301	680.6301
	worst	680.6301	680.6301	680.6301
6	suc.rate	30/30	3/30	30/30
				order= 2
	best	−310.0000	−310.0000	−309.9998
	worst	−310.0000	−152.0000	−309.9998
7	suc.rate	30/30	30/30	0/30
	best	$-3.0666e + 004$	$-3.0666e + 004$	-
	worst	$-3.0666e + 004$	$-3.0666e + 004$	-
8	suc.rate	30/30	21/30	30/30
				order= 4
	best	−5.5080	−5.5080	−5.5079
	worst	−5.5080	−4.0537	−5.5079
9	suc.rate	30/30	23/30	0/30
	best	$7.0492e + 003$	$7.0492e + 003$	-
	worst	$7.0492e + 003$	$1.6855e + 004$	-
10	suc.rate	30/30	2/30	0/30
	best	−575.5925	−575.5925	-
	worst	−575.5925	$2.9359e + 005$	-
11	suc.rate	30/30	0/30	0/30
	best	−1.1078	68.6441	-
	worst	−1.1078	674.2090	-
12	suc.rate	30/30	3/30	0/30
	best	−153.6180	−153.6180	-
	worst	−153.6180	−19.9575	-
13	suc.rate	30/30	30/30	30/30
				order= 3
	best	0.7500	0.7500	0.7500
	worst	0.7500	0.7500	0.7500

continue goes here. . .

Problem	statistic	GOM	Interior Point	GloptiPoly 3
14	suc.rate	30/30	30/30	30/30
	best	-16.7389	-16.7389	-16.7389
	worst	-16.7389	-16.7389	-16.7389
15	suc.rate	30/30	4/30	0/30
	best	-1.0000	-1.0000	-
	worst	-1.0000	$-5.5063e-023$	-

In the table 3, we can see that Interior Point Method can solve Problem 1, 3-5, 7, 13-14 successfully. For Problem 2, 8-9, Interior Point Method performs not bad and the success rate is over 2/3. However for Problem 6, 10-12 and 15, Interior Point Method only succeeds less than 5 times out of 30 or even fails all 30 times for Problem 11.

When we use GloptiPoly 3 to solve non-convex polynomial programming problems, it may not return the global optimum but a lower bound. The default order in GloptiPoly 3 is such that twice the order is greater than or equal to the maximal degree occurring in the polynomial expressions of the original optimization problem. More importantly, the series of optima of SDP-relaxations of increasing orders converges monotonically to the global optimum [25]. However, the computational time increase quickly with the relaxation order and the computer may return 'out of memory' when the order is big enough.

In the table 3, we use the solver GloptiPoly 3 to solve Problem 1-15. We run GloptiPoly 3 30 times for each problem with fixed relaxation order. First, we use the default order to calculate it. If it fails, we increase the order so that the problem can be solved. For example, for Problem 1, GloptiPoly 3 fails to solve it until the order equals to 4. If a problem cannot be solved by the solver GloptiPoly 3 with increasing orders from default order to the order making it out of memory, then success rate is 0/30. From the above table, we can see GloptiPoly 3 solves Problem 1-6, 8, 13-14 successfully. For the rest problems, GloptiPoly 3 failed, and return 'out of memory'.

For the large scale Problem 10-12, the regularization methods for SOS relaxations in large scale polynomial optimization provided in [5] gave global or approximate global optimal values. By our method GOM, we got the same results with those in [5].

Note, all computations in the paper were implemented on a Microsoft Windows XP Desktop of 3.46GB memory and 2.99GHz CPU frequency.

5. Conclusion

We study a necessary and sufficient condition to a point being a global minimizer for a constrained univariate polynomial programming problem. Necessary global optimality conditions for a general constrained polynomial programming problem (*GPP*) are provided based on this necessary and sufficient condition. A new local optimization method is designed according to these necessary global conditions which improve the traditional local optimization method (based on KKT conditions). A new global optimization method is designed by combining the new local optimization method and an auxiliary function. The numerical examples illustrate that our methods are efficient and stable.

References

[1] C.A. Floudas, P.M.A. Pardalos, Collection of test problems for constrained global optimization algorithms, In: Goos, G., Hartmanis, J.(eds.) Lecture Notes in Computer Science. Springer-Verlag. (1990)

[2] S. He, Z. Li, S. Zhang, Approximation algorithms for homogeneous polynomial optimization with quadratic constraints, Math. program. 125(2) (2010) 353-383.

[3] J. Nie, L. Wang, Regularization methods for SDP relaxations in large-scale polynomial optimization, SIAM J. Optim. 22(2) (2012) 408-428.

[4] D. Han, Global optimization with polynomials, http://hdl.handle.net/1721.1/3883. (2004) Accessed 20 December 2013.

[5] J. Nie, Regularization methods for sum of squares relaxations in large scale polynomial optimization, No. arXiv:0909.3551. (2009)

[6] Z.Q. Luo, S. Zhang, A semidefenite relaxation scheme for multivariate quartic polynomial optimization with quadratic constraints, SIAM J. Optim. 20(4) (2010) 1716-1736.

[7] S. Kim, M. Kojima, H. Waki Generalized lagrangian duals and sums of squares relaxations of sparse polynomial optimization problems, SIAM J. Optim. 15(3) (2005) 697-719.

[8] B. JIANG, Z. LI, S. ZHANG, Approximation methods for complex polynomial optimization, Technical report, Department of Industrial and Systems Engineering, University of Minnesota. (2012)

[9] C. Ling, X. Zhang, L. Qi, Semidefinite relaxation approximation for multivariate bi-quadratic optimization with quadratic constraints, Numer. Linear Algebr. Appl. 19(1) (2012) 113-131.

[10] J.B. Hiriart-Urruty, C. Lemarechal, Testing necessary and sufficient conditions for global optimality in the problem of maximizing a convex quadratic funcion over a convex polyhedron, Preliminary Report, University of Paul Sabatier, Toulouse. (1990)

[11] J.B. Hiriart-Urruty, Global optimality conditions in maximizing a convex quadratic function under convex quadratic constraints, J. Glob. Optim. 21(4) (2001) 443-453.

[12] J.M Peng,Y. Yuan, Optimality conditions for the minimization of a quadratic with two quadratic constraints, SIAM J. Optim. 7(3) (1997) 579-594.

[13] V. Jeyakumar, A.M. Rubinow, Z.Y. Wu, Non-convex quadratic minimization problems with quadratic constraints: global optimality conditions, Math. Program. 110(3) (2007) 521-541.

[14] H. Schichl, A. Neumaier, Transposition theorems and qualification-free optimality conditions, SIAM J. Optim. 17(4) (2006) 1035-1055.

[15] J.B. Lasserre, Global optimization with polynomials and the problem of moments, SIAM J. Optim. 11(3) (2001) 796-817.

[16] V. Jeyakumar, G.Y. Li, Necessary global optimality conditions for nonlinear programming problems with polynomial constraints, Math. Program. 126(2) (2011) 393-399.

[17] V. Visweswaran, C.A. Floudas, Unconstrained and constrained global optimization of polynomial functions in one variable, J. Glob. Optim. 2 (1) (1992) 73-99.

[18] H.D. Sherali, C.H. Tuncbilek, New reformulation linearization/convexification relaxations for univariate and multivariate polynomial programming problems£ Oper. Res. Lett. 21(1) (1997) 1-9.

[19] C.A. Floudas, V. Visweswaran, A global optimization algorithm (GOP) for certain classes of nonconvex NLPs-I. theorey, Comput. Chem. Eng. 14(12) (1990) 1397-1417.

[20] S. Fortune, An iterated eigenvalue algorithm for approximating roots of univariate polynomials, J. Symb. Comput. 33(5) (2002) 627-646.

[21] D.A. Bini, L. Gemignani, V.Y. Pan, Inverse power and Durand-Kerner iterations for univariate polynomial root-finding, Comput. Math. Appl. 47(2) (2004) 447-459.

[22] M.S. Bazaraa, H.D. Sherali, C.M. Shetty, Nonlinear Programming: Theory and Algorithms, 3rd edition, John Wiley and Sons, Inc., Hoboken, New Jersey. (2006)

[23] Z.Y Wu, H.W.J. Lee, L.S. Zhang, X.M.Yang, A novel filled function method and quasi-filled function method for global optimization, Comput. Optim. Appl. 34(2) (2006) 249-272.

[24] M. Laguna, R. Marti, Experimental testing of advanced scatter search designs for global optimization of multimodal functions, J. Glob. Optim. 33(2) (2005) 235-255.

[25] D. Henrion, J.B. Lasserre, GloptiPoly: Global optimization over polynomials with Matlab and SeDuMi, ACM Trans. Math. Softw. (TOMS). 29(2) (2003) 165-194.

[26] D. Henrion, J.B. Lasserre, J. Lofberg, GloptiPoly 3: moments, optimization and semidefinite programming, Optim. Methods Softw. 24(4-5) (2009) 761-779.

6. Appendix

Problem 1

$$\begin{aligned}
\min \quad & f(x) := -2x_1 + x_2 - x_3 \\
s.t. \quad & x_1 + x_2 + x_3 \leq 4 \\
& x_1 \leq 2 \\
& x_3 \leq 3 \\
& 3x_2 + x_3 \leq 6 \\
& x_1, x_2, x_3 \geq 0 \\
& x^T B^T B x - 2r^T B x + \|r\|^2 - 0.25\|b - v\|^2 \geq 0
\end{aligned}$$

where

$$\begin{aligned}
B &= \begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ -2 & 1 & -1 \end{bmatrix} \\
b &= [3, 0, -4] \\
v &= [0, -1, -6] \\
r &= [1.5, -0.5, -5]
\end{aligned}$$

14

Problem 2

$$\min \quad f(x) := 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$$

$$s.t. \quad 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leq 0$$
$$2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leq 0$$
$$2x_2 + 2x_3 + x_{11} + x_{12} - 10 \leq 0$$
$$-8x_1 + x_{10} \leq 0$$
$$-8x_2 + x_{11} \leq 0$$
$$-8x_3 + x_{12} \leq 0$$
$$-2x_4 - x_5 + x_{10} \leq 0$$
$$-2x_6 - x_7 + x_{11} \leq 0$$
$$-2x_8 - x_9 + x_{12} \leq 0$$
$$x_i \geq 0, \ i = 1, \cdots, 13$$
$$x_i \leq 1, \ i = 1, \cdots, 9, 13$$
$$x_i \leq 100, \ i = 10, \cdots, 12.$$

Problem 3

$$\min \quad f(x) := x_1^2 + x_2^2 + x_1 x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + \ldots$$
$$4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 + \ldots$$
$$7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45;$$

$$s.t. \quad 4x_1 + 5x_2 - 3x_7 + 9x_8 - 105 \leq 0$$
$$10x_1 - 8x_2 - 17x_7 + 2x_8 \leq 0$$
$$-8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \leq 0$$
$$3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \leq 0$$
$$5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \leq 0$$
$$0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 \leq 0$$
$$x_1^2 + 2(x_2 - 2)^2 - 2x_1 x_2 + 14x_5 - 6x_6 \leq 0$$
$$-3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \leq 0$$
$$-10 \leq x_i \leq 10, \ i = 1, \cdots, 10.$$

Problem 4

$$\min \quad f(x) := (x_1 - 10)^3 + (x_2 - 20)^3$$
$$s.t. \quad -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \leq 0$$
$$(x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0$$
$$13 \leq x_1 \leq 100, 0 \leq x_2 \leq 100.$$

Problem 5

$$\min \quad f(x) := (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + \ldots$$
$$10x_5^6 + 7x_6^2 + x_7^4 - 4x_6 x_7 - 10x_6 - 8x_7;$$
$$s.t. \quad v1 + 3v2^2 + x_3 + 4x_4^2 + 5x_5 - 127 \leq 0$$
$$7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 - 282 \leq 0$$

$$23x_1 + v2 + 6x_6^2 - 8x_7 - 196 \leq 0$$
$$2v1 + v2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \leq 0$$
$$-10 \leq x_i \leq 10, \ i = 1, \cdots, 7$$
where $\ v1 = 2x_1^2, \ v2 = x_2^2.$

Problem 6

$$\min \quad f(x) := -25(x_1 - 2)^2 - (x_2 - 2)^2 - (x_3 - 1)^2$$
$$-(x_4 - 4)^2 - (x_5 - 1)^2 - (x_6 - 4)^2$$
$$s.t. \quad (x_3 - 3)^2 + x_4 \geq 4$$
$$(x_5 - 3)^2 + x_6 \geq 4$$
$$x_1 - 3x_2 \leq 2$$
$$-x_1 + x_2 \leq 2$$
$$x_1 + x_2 \leq 6$$
$$x_1 + x_2 \geq 2$$
$$0 \leq x_1 \leq 6$$
$$0 \leq x_2 \leq 8$$
$$1 \leq x_3 \leq 5$$
$$0 \leq x_4 \leq 6$$
$$1 \leq x_5 \leq 5$$
$$0 \leq x_6 \leq 10.$$

Problem 7

$$\min \quad f(x) := 37.293239x_1 + 0.8356891x_1x_5 + 5.3578547x_3^2 - 40792.141$$
$$s.t. \quad -0.0022053x_3x_5 + 0.0056858x_2x_5 + 0.0006262x_1x_4 - 6.665593 \leq 0$$
$$0.0022053x_3x_5 - 0.0056858x_2x_5 - 0.0006262x_1x_4 - 85.334407 \leq 0$$
$$0.0071317x_2x_5 + 0.0021813x_3^2 + 0.0029955x_1x_2 - 29.48751 \leq 0$$
$$-0.0071317x_2x_5 - 0.0021813x_3^2 - 0.0029955x_1x_2 + 9.48751 \leq 0$$
$$0.0047026x_3x_5 + 0.0019085x_3x_4 + 0.0012547x_1x_3 - 15.699039 \leq 0$$
$$-0.0047026x_3x_5 - 0.0019085x_3x_4 - 0.0012547x_1x_3 + 10.699039 \leq 0$$
$$78 \leq x_1 \leq 102$$
$$33 \leq x_2 \leq 45$$
$$27 \leq x_3 \leq 45$$
$$27 \leq x_4 \leq 45$$
$$27 \leq x_5 \leq 45.$$

Problem 8

$$\min \quad f(x) := -x - y$$
$$s.t. \quad y \leq 2x^4 - 8x^3 + 8x^2 + 2$$
$$y \leq 4x^4 - 32x^3 + 88x^2 - 96x + 36$$
$$0 \leq x \leq 3$$
$$0 \leq y \leq 4.$$

16

Problem 9

$$\min \quad f(x) := x_1 + x_2 + x_3$$
$$s.t. \quad -1 + 0.0025(x_4 + x_6) \le 0$$
$$-1 + 0.0025(-x_4 + x_5 + x_7) \le 0$$
$$-1 + 0.01(-x_5 + x_8) \le 0$$
$$100x_1 - x_1 x_6 + 833.33252 x_4 - 83333.333 \le 0$$
$$x_2 x_4 - x_2 x_7 - 1250 x_4 + 1250 x_5 \le 0$$
$$x_3 x_5 - x_3 x_8 - 2500 x_5 + 1250000 \le 0$$
$$l_i \le x_i \le u_i, \ i = 1, \cdots, 8$$
$$\text{where} \quad l = 10 \times (10, 100, 100, 1, 1, 1, 1, 1)$$
$$u = 1000 \times (10, 10, 10, 1, 1, 1, 1, 1).$$

Problem 10

$$\min \quad \sum_{1 \le i < j < k \le n} (i+j)x_i x_j x_k + (j+k)x_i^2 x_j^2 x_k^2$$
$$s.t. \quad x_1^4 + \cdots + x_n^4 \le 1$$
$$\text{where} \quad n = 15.$$

Problem 11

$$\min \quad \sum_{1 \le i < j < k \le n} x_i x_j x_k (1 + x_i + x_j + x_k) + i x_i^6 + j x_j^6 + k x_k^6$$
$$s.t. \quad x_1^4 + \cdots + x_{\frac{n}{2}}^4 \le 1$$
$$s.t. \quad x_{\frac{n}{2}+1}^4 + \cdots + x_n^4 \le 1$$
$$\text{where} \quad n = 16.$$

Problem 12

$$\min \quad \sum_{1 \le i < j < k \le \frac{n}{2}} i x_i x_j x_k + j x_{\frac{n}{2}+i} x_{\frac{n}{2}+j} x_{\frac{n}{2}+k} + k x_i x_j x_k x_{\frac{n}{2}+i} x_{\frac{n}{2}+j} x_{\frac{n}{2}+k}$$
$$s.t. \quad x_1^4 + \cdots + x_{\frac{n}{2}}^4 \le 1$$
$$x_{\frac{n}{2}+1}^4 + \cdots + x_n^4 \le 1$$
$$\text{where} \quad n = 20.$$

Problem 13

$$\min \quad f(x) := x_1^2 + (x_2 - 1)^2$$
$$s.t. \quad x_2 - x_1^2 = 0$$
$$-1 \le x_i \le 1, \ i = 1, 2.$$

Problem 14

$$\min \quad f(x) := -12x_1 - 7x_2 + x_2^2$$
$$s.t. \quad -2x_1^4 + 2 - x_2 = 0$$
$$0 \le x_1 \le 2, \ 0 \le x_2 \le 3.$$

Problem 15

$$\min \quad f(x) := (\sqrt{n})^n \prod_{i=1}^{n} x_i$$

$$s.t. \quad \sum_{i=1}^{n} x_i^2 - 1 = 0$$

$$0 \le x_i \le 1, \ i = 1, \cdots, n$$

where $\quad n = 20.$