数理解析研究所講究録
第 1930 巻 2015 年 39-50

Towards Calculating Approximate GCD of Univariate Polynomials with Semidefinite Programming

照井 章

AKIRA TERUI

筑波大学 数理物質科学研究科

GRADUATE SCHOOL OF PURE AND APPLIED SCIENCES, UNIVERSITY OF TSUKUBA *

Abstract

We consider calculating approximate greatest common divisor (GCD) of univariate polynomials with semidefinite programming (SDP). In our method for calculating approximate GCD, the derived constrained optimization problem has nonconvex constraints in general, which may cause a difficulty for finding proper perturbations for the input polynomials with finding an approximate GCD of the given degree. On the other hand, SDP is a convex programming and widely used in approximating or relaxing nonconvex programs into convex ones for seeking better optimizer, thus SDP will be useful for solving various problems in symbolic-numeric computation including an approximate GCD problem. In this paper, we show attempts for calculating approximate GCD with Lasserre's SDP relaxation and an SDP relaxation of quadratic constrained quadratic programming (QCQP) using an SDP solver SDPA by an example.

1 Introduction

For algebraic computations on polynomials and matrices, approximate algebraic algorithms are attracting broad range of attentions recently. These algorithms take inputs with some "noise" such as polynomials with floating-point number coefficients with rounding errors, or more practical errors such as measurement errors, then, with minimal changes on the inputs, seek a meaningful answer that reflect desired property of the input, such as a common factor of a given degree. By this characteristic, approximate algebraic algorithms are expected to be applicable to more wide range of problems, especially those to which exact algebraic algorithms were not applicable.

As an approximate algebraic algorithm, we consider calculating the approximate greatest common divisor (GCD) of univariate polynomials, such that, for a given pair of polynomials and a degree d, finding a pair of polynomials which has a GCD of degree d and whose coefficients are perturbations from those in the original inputs, with making the perturbations as small as possible, along with the GCD. This problem has been extensively studied with various approaches including the Euclidean method on the polynomial remainder sequence (PRS) ([2], [18], [19]), the singular value decomposition (SVD) of the Sylvester matrix ([4], [6]), the QR factorization of the Sylvester matrix or its displacements ([5], [27], [30]), Padé approximation [15], optimization strategies ([3], [8], [9], [11], [13], [28]). Furthermore, stable methods for ill-conditioned problems have been discussed ([5], [14], [17]).

*terui@math.tsukuba.ac.jp

Among methods in the above, we focus our attention on optimization strategies. Already proposed algorithms utilize iterative methods including the Levenberg-Marquardt method [3], the Gauss-Newton method [28] and the structured total least norm (STLN) method ([8], [9]). Among them, STLN-based methods have shown good performance calculating approximate GCD with sufficiently small perturbations efficiently.

Recently, the present author has proposed a method called GPGCD ([21], [22], [23]). This is an iterative method with transferring the original approximate GCD problem into a constrained optimization problem, then solving it by the so-called modified Newton method [20], which is a generalization of the gradient-projection method [16]. We have shown that our method calculates approximate GCD with perturbations as small as those calculated by the STLN-based methods and with significantly better efficiency than theirs.

However, we may face a difficulty that, from a given pair of polynomials, we cannot calculate proper perturbation so that the given polynomials have an approximate GCD of the given degree. One reason is that the derived optimization problem has nonconvex constraints which will be very hard to solve in general, and the GPGCD method depends on Newton-like iterative method with local convergence. Thus, in such a case that the pair of input polynomials lies too far away from their approximate GCDs of the given degree, we may fail to find an approximate GCD.

In our setting of the approximate GCD problem, calculation of approximate GCD depends on how we can find a good answer for derived polynomial optimization problem (POP). In this paper, we consider calculating approximate GCD with semidefinite programming (SDP) for finding better optimizers. SDP is a convex optimization programming, thus, by approximating or relaxing a non-convex optimization problem including our approximate GCD problems with SDP, we can expect to obtain satisfactory answers which have never been obtained by our previous methods. In the context of symbolic-numeric computation, while recent research results include the sum of squares (SOS) relaxation and certification of global optimization problems for polynomials ([7], [10], [13]), we focus our attention on two SDP relaxations: one is Lasserre's relaxation for general POP [12] and the other is relaxation of quadratically constrained quadratic programming (QCQP). We show examples of our attempts for calculation of approximate GCD by these relaxations with an SDP solver SDPA [26].

The rest part of the paper is organized as follows. In Section 2, we review how to transform the approximate GCD problem into a constrained minimization problem as in the GPGCD method [22]. In Section 3, we review SDP relaxations for general POP by Lasserre and QCQP. In Section 4, we show results of computations of our attempt for calculation of approximate GCD with an SDP solver SDPA for both relaxations by examples. In Section 5, we discuss future direction of research for overcoming problems in computing approximate GCD by SDP according to the results in the previous section.

2 Formulation of the Approximate GCD Problem

Let $F(x)$ and $G(x)$ be univariate polynomials with the real or the complex coefficients, given as

$$F(x) = f_m x^m + f_{m-1} x^{m-1} + \cdots + f_0, \quad G(x) = g_n x^n + g_{n-1} x^{n-1} + \cdots + g_0,$$

with $m \geq n > 0$. We permit F and G to be relatively prime in general. For a given integer d satisfying $n \geq d > 0$, let us calculate a deformation of $F(x)$ and $G(x)$ in the form of

$$\begin{aligned} \tilde{F}(x) &= \tilde{f}_m x^m + \cdots + \tilde{f}_0 x^0 = F(x) + \Delta F(x) = H(x) \cdot \bar{F}(x), \\ \tilde{G}(x) &= \tilde{g}_n x^n + \cdots + \tilde{g}_0 x^0 = G(x) + \Delta G(x) = H(x) \cdot \bar{G}(x), \end{aligned} \tag{1}$$

where $\Delta F(x)$, $\Delta G(x)$ are polynomials whose degrees do not exceed those of $F(x)$ and $G(x)$, respectively, $H(x)$ is a polynomial of degree d, and $\bar{F}(x)$ and $\bar{G}(x)$ are pairwise relatively prime. If we find $\tilde{F}$, $\tilde{G}$, $\bar{F}$, $\bar{G}$ and H satisfying (1), then we call H *an approximate GCD of F and G*. For a given degree d, we tackle the problem of finding an approximate GCD H with minimizing the norm of the deformations $\|\Delta F(x)\|_2^2 + \|\Delta G(x)\|_2^2$.

In the case $\tilde{F}(x)$ and $\tilde{G}(x)$ have a GCD of degree d, then the theory of subresultants tells us that the $(d-1)$-th subresultant of $\tilde{F}$ and $\tilde{G}$ becomes zero, namely we have $\mathrm{S}_{d-1}(\tilde{F}, \tilde{G}) = 0$, where $\mathrm{S}_k(\tilde{F}, \tilde{G})$ denotes the subresultant of $\tilde{F}$ and $\tilde{G}$ of degree k. Then, the $(d-1)$-th subresultant matrix

$$N_{d-1}(\tilde{F}, \tilde{G}) = \underbrace{\begin{pmatrix} \tilde{f}_m & & & \tilde{g}_n & & \\ \vdots & \ddots & & \vdots & \ddots & \\ \tilde{f}_0 & & \tilde{f}_m & \tilde{g}_0 & & \tilde{g}_n \\ & \ddots & \vdots & & \ddots & \vdots \\ & & \tilde{f}_0 & & & \tilde{g}_0 \end{pmatrix}}_{n-d+1 \qquad\qquad m-d+1}, \tag{2}$$

where the k-th subresultant matrix $N_k(\tilde{F}, \tilde{G})$ is a submatrix of the Sylvester matrix $N(\tilde{F}, \tilde{G})$ by taking the left $n-k$ columns of coefficients of $\tilde{F}$ and the left $m-k$ columns of coefficients of $\tilde{G}$, has a kernel of dimension equal to 1. Thus, there exist polynomials $A(x), B(x) \in \mathbb{R}[x]$ or $\mathbb{C}[x]$ satisfying

$$A\tilde{F} + B\tilde{G} = 0, \tag{3}$$

with $\deg(A) < n-d$ and $\deg(B) < m-d$ and $A(x)$ and $B(x)$ are relatively prime. Therefore, for the given $F(x)$, $G(x)$ and d, our problem is to find $\Delta F(x)$, $\Delta G(x)$, $A(x)$ and $B(x)$ satisfying Eq. (3) with making $\|\Delta F\|_2^2 + \|\Delta G\|_2^2$ as small as possible.

Assuming that we have $F(x)$ and $G(x)$ as polynomials with the real coefficients and find an approximate GCD with the real coefficients as well, we represent $A(x)$ and $B(x)$ with the real coefficients as

$$A(x) = a_{n-d}x^{n-d} + \cdots + a_0 x^0, \quad B(x) = b_{m-d}x^{m-d} + \cdots + b_0 x^0, \tag{4}$$

respectively, thus $\|\Delta F\|_2^2 + \|\Delta G\|_2^2$ and Eq. (3) become as

$$\|\Delta F\|_2^2 + \|\Delta G\|_2^2 = (\tilde{f}_m - f_m)^2 + \cdots + (\tilde{f}_0 - f_0)^2 + (\tilde{g}_n - g_n)^2 + \cdots + (\tilde{g}_0 - g_0)^2, \tag{5}$$

$$N_{d-1}(\tilde{F}, \tilde{G}) \cdot \boldsymbol{v} = \boldsymbol{0}, \tag{6}$$

respectively, with $N_{d-1}(\tilde{F}, \tilde{G})$ as in (2) and

$$\boldsymbol{v} = {}^t(a_{n-d}, \cdots, a_0, b_{m-d}, \cdots, b_0). \tag{7}$$

Then, Eq. (6) is regarded as a system of $m+n-d+1$ equations in $\tilde{f}_m, \ldots, \tilde{f}_0$, $\tilde{g}_n, \ldots, \tilde{g}_0$, $a_{n-d}, \ldots, a_0$, $b_{m-d}, \ldots, b_0$, as

$$q_1 = \tilde{f}_m a_{n-d} + \tilde{g}_n b_{m-d} = 0, \ldots, q_{m+n-d+1} = \tilde{f}_0 a_0 + \tilde{g}_0 b_0 = 0, \tag{8}$$

by putting q_j as the j-th row. Furthermore, for solving the problem below stably, we add another constraint enforcing the coefficients of $A(x)$ and $B(x)$ such that $\|A(x)\|_2^2 + \|B(x)\|_2^2 = 1$; thus we add

$$q_0 = a_{n-d}^2 + \cdots + a_0^2 + b_{m-d}^2 + \cdots + b_0^2 - 1 = 0 \tag{9}$$

into Eq. (8).

Now, we substitute the variables

$$(\tilde{f}_m, \ldots, \tilde{f}_0, \tilde{g}_n, \ldots, \tilde{g}_0, a_{n-d}, \ldots, a_0, b_{m-d}, \ldots, b_0) \tag{10}$$

as

$$\boldsymbol{x} = (x_1, \ldots, x_{2(m+n-d+2)}), \tag{11}$$

thus Eq. (5) and (8) with (9) become

$$f(\boldsymbol{x}) = (x_1 - f_m)^2 + \cdots + (x_{m+1} - f_0)^2 + (x_{m+2} - g_n)^2 + \cdots + (x_{m+n+2} - g_0)^2, \tag{12}$$

$$\boldsymbol{q}(\boldsymbol{x}) = {}^t(q_0(\boldsymbol{x}), q_1(\boldsymbol{x}), \ldots, q_{m+n-d+1}(\boldsymbol{x})) = \boldsymbol{0}, \tag{13}$$

respectively. Therefore, the problem of finding an approximate GCD can be formulated as a constrained minimization problem of finding a minimizer of the objective function $f(\boldsymbol{x})$ in (12), subject to $\boldsymbol{q}(\boldsymbol{x}) = \boldsymbol{0}$ in Eq. (13).

3 Semidefinite Programming

Semidefinite programming (SDP) can be regarded as a generalization of linear programming (LP). Let S^n be a set of the $n \times n$ real symmetric matrices, and let $A_i \in S^n$ $(i = 1, \ldots, m)$ be constant matrices, $X \in S^n$ be a variable matrix, $b_i \in \mathbb{R}$ $(i = 1, \ldots, m)$ be constants. Then, a standard form of SDP is denoted as

$$\min \quad A_0 \bullet X \quad \text{s.t.} \quad A_i \bullet X = b_i, \quad i = 1, \ldots, m, \quad X \succeq \boldsymbol{O},$$

where, for $U, V \in S^n$, $U \bullet V$ denotes an inner product of U and V as $\sum_{i=1}^n \sum_{j=1}^n U_{ij} V_{ij}$.

Let us refer to the above problem as a primal SDP. Then, an analogous to LP, a dual SDP can denoted as

$$\max \quad \sum_{i=1}^m b_i z_i \quad \text{s.t.} \quad \sum_{i=1}^m A_i z_i + Y = A_0, \quad Y \succeq \boldsymbol{O},$$

where $z_i \in \mathbb{R}$ and $Y \in S^n$.

SDP is a special class of convex programming which includes LP and (convex) quadratically constrained quadratic programming (QCQP). With this and other reasons [24], there arise active research in various optimization problems by approximating or relaxing them into SDP to search optimizer effectively and efficiently. We review two types of SDP relaxation of POP as follows.

3.1 Lasserre's SDP relaxation

In this section, we review Lasserre's SDP relaxation for general POP [12]. For $\boldsymbol{x} = (x_1, \dots, x_n) \in \mathbb{R}^n$ and $f_0, \dots, f_m \in \mathbb{R}[x_1, \dots, x_n]$, let us consider an optimization problem:

$$p^* = \min f_0(\boldsymbol{x}) \quad \text{s.t.} \quad f_j(\boldsymbol{x}) \geq 0, \quad j = 1, \dots, m. \tag{14}$$

For $j = 0, 1, \dots, m$, let $d_j = \lceil \deg(f_j)/2 \rceil$. For given positive integer d, let $\boldsymbol{u}_d$ be a $\binom{n+d}{d}$-dimensional row vector defined as

$$\boldsymbol{u}_d(\boldsymbol{x}) = (1, x_1, \dots, x_n, x_1^2, x_1 x_2, \dots, x_1^d, \dots, x_n^d).$$

Then, for any integer N satisfying $N \geq \max_{j=0,\dots,m} d_j$, define

$$M_N(\boldsymbol{x}) = ({}^t\boldsymbol{u}_N(\boldsymbol{x}))\, \boldsymbol{u}_N(\boldsymbol{x}), \quad f_j(\boldsymbol{x}) M_{N-d_j}(\boldsymbol{x}) = f_j(\boldsymbol{x})\, ({}^t\boldsymbol{u}_{N-d_j}(\boldsymbol{x}))\, \boldsymbol{u}_{N-d_j}(\boldsymbol{x}).$$

Note that total degree of the term in the element of $f_j(\boldsymbol{x}) M_{N-d_j}(\boldsymbol{x})$ does not exceed $2N$. Then, we see that $M_N(\boldsymbol{x}) \succeq \boldsymbol{O}$ for any $x \in \mathbb{R}^n$ and $f_j(\boldsymbol{x}) M_{N-d_j}(\boldsymbol{x}) \succeq \boldsymbol{O}$ for $\boldsymbol{x}$ satisfying (14).

In the next step, we "linearize" variable $\boldsymbol{x}$ with $\boldsymbol{y}_{\boldsymbol{a}}$, as follows. For nonnegative integers $a_1, \dots, a_n$, let $\boldsymbol{a} = (a_1, \dots, a_n)$ and denote $x_1^{a_1} \cdots x_n^{a_n}$ to $\boldsymbol{x}^{\boldsymbol{a}}$. Then, let us express a monomial $\boldsymbol{x}^{\boldsymbol{a}}$ with a "linearized" variable $\boldsymbol{y}_{\boldsymbol{a}}$. For $\boldsymbol{x}^{\boldsymbol{a}}$ belonging to the support of f, or $\mathrm{supp}(f) = \{\boldsymbol{a} \mid c_{\boldsymbol{a}} \neq 0 \text{ for } f(\boldsymbol{x}) = \sum_{\boldsymbol{a} \in \mathbb{Z}_+^n} c_{\boldsymbol{a}} \boldsymbol{x}^{\boldsymbol{a}}\}$, define $\boldsymbol{y} = \{\boldsymbol{y}_{\boldsymbol{a}} \mid \boldsymbol{a} \in \mathrm{supp}(f),\ \boldsymbol{a} \neq (1, \dots, 0)\}$. Note that, in the linearized parameter $\boldsymbol{y}_{\boldsymbol{a}}$, we transform $\boldsymbol{x}^{\boldsymbol{0}}$ to 1 thus $\boldsymbol{y}_{\boldsymbol{0}} = 1 \notin \boldsymbol{y}$. Then, for $f_j(\boldsymbol{x}) M_{N-d_j}(\boldsymbol{x})$ expressed as

$$f_j(\boldsymbol{x}) M_{N-d_j}(\boldsymbol{x}) = A_0^j + \sum_{\boldsymbol{a}} \boldsymbol{x}^{\boldsymbol{a}} A_{\boldsymbol{a}}^j,$$

where A_0^j is a matrix with only $(1,1)$ element is nonzero and $A_{\boldsymbol{a}}^j$ is a symmetric matrix, we obtain

$$f_j(\boldsymbol{y}) M_{N-d_j}(\boldsymbol{y}) = A_0^j + \sum_{\boldsymbol{a}} \boldsymbol{y}_{\boldsymbol{a}} A_{\boldsymbol{a}}^j.$$

Furthermore, for $M_N(\boldsymbol{x})$ expressed as

$$M_N(\boldsymbol{x}) = E_0 + \sum_{\boldsymbol{a}} \boldsymbol{x}^{\boldsymbol{a}} E_{\boldsymbol{a}},$$

where E_0 is a matrix with $(E_0)_{1,1} = 1$ and the other element is equal to 0 and $E_{\boldsymbol{a}}$ is a matrix of 0 or 1 entries, we obtain

$$M_N(\boldsymbol{y}) = E_0 + \sum_{\boldsymbol{a}} \boldsymbol{y}_{\boldsymbol{a}} E_{\boldsymbol{a}}.$$

With the above transformation, we obtain an SDP as

$$\min \quad f_0(\boldsymbol{y}) \quad \text{s.t.} \quad E_0 + \sum_{\boldsymbol{a}} \boldsymbol{y}^{\boldsymbol{a}} E_{\boldsymbol{a}} \succeq \boldsymbol{O}, \quad A_0^j + \sum_{\boldsymbol{a}} \boldsymbol{y}_{\boldsymbol{a}} A_{\boldsymbol{a}}^j \cdot \succeq \boldsymbol{O}, \tag{15}$$

which is a relaxation of (14) with $\boldsymbol{y}_{\boldsymbol{a}} = x^{\boldsymbol{a}}$.

3.2 SDP Relaxation of Quadratically Constrained Quadratic Programming (QCQP)

The following relaxation is known as Shor's relaxation scheme [29]. We consider the following form of quadratically constrained quadratic programming (QCQP) problem:

$$\begin{array}{ll} \min & f_0(\boldsymbol{x}) = \boldsymbol{x}^T A_0 \boldsymbol{x} + b_0^T \boldsymbol{x} \\ \text{s.t.} & f_i(\boldsymbol{x}) = \boldsymbol{x}^T A_i \boldsymbol{x} + b_i^T \boldsymbol{x} + c_i \le 0, \quad i \in \mathcal{I}, \\ & g_i(\boldsymbol{x}) = \boldsymbol{x}^T A_i \boldsymbol{x} + b_i^T \boldsymbol{x} + c_i = 0, \quad i \in \mathcal{E}, \end{array}$$

where $A_i \in S^n$, $\boldsymbol{x} \in \mathbb{R}^n$ and $\mathcal{I}$ and $\mathcal{E}$ are sets of indices. Then, an SDP relaxation can be obtained as

$$\begin{array}{ll} \min & A_0 \bullet X + b_0^T \boldsymbol{x} \\ \text{s.t.} & A_i \bullet X + b_i^T \boldsymbol{x} + c_i \le 0, \quad i \in \mathcal{I}, \\ & A_i \bullet X + b_i^T \boldsymbol{x} + c_i = 0, \quad i \in \mathcal{E}, \\ & \begin{pmatrix} 1 & \boldsymbol{x}^T \\ \boldsymbol{x} & X \end{pmatrix} \succeq O, \end{array}$$

where $X = \boldsymbol{x}\boldsymbol{x}^T$.

4 SDP Relaxations of Approximate GCD Problem by an Example

In this section, we show how SDP relaxations in the above can be applied to our approximate GCD problem by an example.

4.1 An Example Problem

Here is an example of approximate GCD problem with input polynomials $F(x)$ and $G(x)$ as

$$\begin{aligned} F(x) &= (x + 2.5)(x + 1.2) + 0.03x - 0.01 = x^2 + 3.73x + 2.99, \\ G(x) &= x + 2.6, \end{aligned}$$

and the degree of the GCD as $d = 1$.

With our formulation, we seek polynomials $\tilde{F}(x) = \tilde{f}_2 x^2 + \tilde{f}_1 x + \tilde{f}_0$, $\tilde{G}(x) = \tilde{g}_1 x + \tilde{g}_0$, $\tilde{A}(x) = \tilde{a}_0$, $\tilde{B}(x) = \tilde{b}_1 x + \tilde{b}_0$ satisfying (3), which can also be expressed as

$$\begin{pmatrix} \tilde{f}_2 & \tilde{g}_1 & \\ \tilde{f}_1 & \tilde{g}_0 & \tilde{g}_1 \\ \tilde{f}_0 & & \tilde{g}_0 \end{pmatrix} \begin{pmatrix} \tilde{a}_0 \\ \tilde{b}_1 \\ \tilde{b}_0 \end{pmatrix} = \mathbf{0},$$

using $N_0(\tilde{F}, \tilde{G})$ as in (2). By adding a constraint on the norm of $\tilde{A}(x)$ and $\tilde{B}(x)$ as in (9) and substituting the variables as

$$\boldsymbol{x} = (x_1, \ldots, x_8) = (\tilde{f}_2, \tilde{f}_1, \tilde{f}_0, \tilde{g}_1, \tilde{g}_0, \tilde{a}_0, \tilde{b}_1, \tilde{b}_0), \tag{16}$$

as in (10) and (11), then we have an objective function $f(\boldsymbol{x})$ as

$$f(\boldsymbol{x}) = (x_1 - 1)^2 + (x_2 - 3.73)^2 + (x_3 - 2.99)^2 + (x_4 - 1)^2 + (x_5 - 2.6)^2, \quad (17)$$

and constraints as

$$\begin{aligned} q_0(\boldsymbol{x}) &= x_6^2 + x_7^2 + x_8^2 - 1 = 0, & q_1(\boldsymbol{x}) &= x_1x_6 + x_4x_7 = 0, \\ q_2(\boldsymbol{x}) &= x_2x_6 + x_5x_7 + x_4x_8 = 0, & q_3(\boldsymbol{x}) &= x_3x_6 + x_5x_8 = 0. \end{aligned} \quad (18)$$

With the GPGCD method, we have calculated $\tilde{F}(x)$ and $\tilde{G}(x)$ as

$$\begin{aligned} \tilde{F}(x) &= x^2 + 3.751137011754 96x + 3.00438744138066, \\ \tilde{G}(x) &= x + 2.59206678142237, \end{aligned}$$

with an approximate GCD as $\tilde{G}(x)$ and the sum of squares of the 2-norm of perturbation as $3.905693416 \times 10^{-5}$.

4.2 SDPA: an SDP Solver

SDPA is an SDP solver "based on Mehrotra-type predictor-corrector infeasible primal-dual interior-point method [26]." It handles the standard form SDP and its dual, and is implemented in C++ using the LAPACK [1] for matrix computations.

SDPA is designed to solve SDP with the following primal ($\mathcal{P}$) and dual ($\mathcal{D}$) input forms, respectively:

$$\mathcal{P}: \quad \min \quad \sum_{i=1}^{m} c_i x_i \quad \text{s.t.} \quad X = \sum_{i=1}^{m} F_i x_i - F_0, \quad X \succeq \boldsymbol{O}, \quad (19)$$

$$\mathcal{D}: \quad \max \quad F_0 \bullet Y \quad \text{s.t.} \quad F_i \bullet Y = c_i, \quad i = 1, \ldots, m, \quad Y \succeq \boldsymbol{O}, \quad (20)$$

where $\boldsymbol{c} = (c_1, \ldots, c_m)^T$ is a cost vector and $\boldsymbol{x} = (x_1, \ldots, x_m)^T$ is a variable vector, $X \in S^n$, $Y \in S^n$ are variable matrices of dimension $n \times n$.

4.3 Lasserre's SDP Relaxation

We have tested the relaxation with $N = 1$ and 2.

In the case of $N = 1$, we have defined

$$\boldsymbol{u}_1(\boldsymbol{x}) = (1, x_1, \ldots, x_8), \quad M_1(\boldsymbol{x}) = \boldsymbol{u}_1(\boldsymbol{x})^T \boldsymbol{u}_1(\boldsymbol{x}),$$

and

$$\boldsymbol{u}_0(\boldsymbol{x}) = \begin{pmatrix} 1 \end{pmatrix}, \quad q_j(\boldsymbol{x}) M_0(\boldsymbol{x}) = \begin{pmatrix} q_j(\boldsymbol{x}) \end{pmatrix},$$

for $j = 0, 1, 2, 3$. Thus, by following Lasserre's SDP relaxation, we have obtained the SDP problem as in (15) with

$$M_1(\boldsymbol{x}) = E_0 + \sum_{\boldsymbol{a}} \boldsymbol{y}^{\boldsymbol{a}} E_{\boldsymbol{a}} \succeq \boldsymbol{O}, \quad f_j(\boldsymbol{y}) M_0(\boldsymbol{y}) = \begin{pmatrix} 1 \end{pmatrix} + \sum_{\boldsymbol{a}} c_{\boldsymbol{a}} \boldsymbol{y}_{\boldsymbol{a}} \begin{pmatrix} 1 \end{pmatrix},$$

for $j = 0, 1, 2, 3$, with E_0 and $E_{\boldsymbol{a}}$ are defined as in the above, and $\boldsymbol{a} = (i_1, \ldots, i_8)$ with $i_j \geq 0$ and $i_1 + \cdots i_8 = 2$. Furthermore, since the constraints in our problem are equality constraints, we have added constraints $f_{4+j}(\boldsymbol{y}) = -f_j(\boldsymbol{y})$ for $j = 0, \cdots, 3$, thus we have 9 constraints in total.

To make an SDPA input, we have to combine several constraints into one, which can be done simply by making $\bar{E}_0 = \mathrm{diag}(E_0, A_0^1, \ldots, A_0^9)$ and $\bar{A}_{\boldsymbol{a}} = \mathrm{diag}(E_{\boldsymbol{a}}, A_{\boldsymbol{a}}^1, \ldots, A_{\boldsymbol{a}}^9)$, where $\mathrm{diag}(A_1, \ldots, A_n)$ denotes the diagonal matrix whose diagonal blocks consist of $A_1, \ldots, A_n$. Then we put $F_0 = -\bar{E}_0$, $F_i = \bar{A}_{\boldsymbol{a}}$ and $c_i = c_{\boldsymbol{a}}$ in (19). A relaxation with $N = 2$ can also be obtained similarly.

In the case of degree 1 relaxation, we have obtained an SDP problem with 44 variables and matrices of dimension 15 with each of which have only few elements. In the case of degree 2 relaxation, we have obtained an SDP problem with 200 variables and matrices of dimension 117.

Unfortunately, in both cases of SDPA calculation, the iterations have stopped in 3 times for degree 1 relaxation and 7 times for degree 2 relaxation, reporting "pdINF", which means there is a possibility that the primal problem (19) and/or the dual problem (20) is infeasible. In the case of degree 1 relaxation, the calculated minimum value for primal optimization problem was $-4.4273467134781560 \times 10^4$. In the case of degree 2 relaxation, the calculated minimum value for primal optimization problem was $-4.2769453981338473 \times 10^{-1}$.

4.4 SDP Relaxation of QCQP

We see that our formulation of approximate GCD problem is a QCQP, thus we have tried to solve its SDP relaxation by SDPA.

Let

$$\boldsymbol{y} = (1, x_1, \ldots, x_8), \quad Y = \boldsymbol{y}\boldsymbol{y}^T, \tag{21}$$

where $x_1, \ldots, x_8$ are defined as in (16), we can express the optimization problem as

$$\min \quad f(\boldsymbol{x}) = A \bullet Y \quad \text{s.t.} \quad q_j(\boldsymbol{x}) = B_j \bullet Y = 0, \quad j = 0, \ldots, 4, \tag{22}$$

where $f(\boldsymbol{x})$ and $q_j(\boldsymbol{x})$ are defined as in (17) and (18), respectively, and A and B_j are real symmetric matrices.

Expressing the approximate GCD problem in (22) has several advantages over Lasserre's SDP relaxation such as that the same problem can be expressed with fewer number of variables. In this case, we set 9×9 matrices as follows:

- A variable matrix $Y = (x_i x_j)_{ij}$ for $i = 0, \ldots, 8$ and $j = 0, \ldots, 8$ with $x_0 = 1$;
- Coefficient matrices as:
  - $A = (a_{ij})$ with 16 nonzero elements;
  - $B_1 = (b_{ij}^{(1)})$ with $b_{7,7}^{(1)} = b_{8,8}^{(1)} = b_{9,9}^{(1)} = 1$;
  - $B_2 = (b_{ij}^{(2)})$ with $b_{2,7}^{(2)} = b_{7,2}^{(2)} = b_{5,8}^{(2)} = b_{8,5}^{(2)} = 1/2$;
  - $B_3 = (b_{ij}^{(3)})$ with $b_{3,7}^{(3)} = b_{7,3}^{(3)} = b_{6,8}^{(3)} = b_{8,6}^{(3)} = b_{5,9}^{(3)} = b_{9,5}^{(3)} = 1/2$;
  - $B_4 = (b_{ij}^{(4)})$ with $b_{4,7}^{(4)} = b_{7,4}^{(4)} = b_{6,9}^{(4)} = b_{9,6}^{(4)} = 1/2$.

$$
Y = \begin{pmatrix}
1.0 & 2.0 & 7.46 & 5.98 & 2.0 & 5.2 & 0.0 & 0.0 & 0.0 \\
2.0 & 4.0 & 14.92 & 11.96 & 4.0 & 10.40 & 0.0 & 0.0 & 0.0 \\
7.46 & 14.92 & 55.65 & 44.61 & 14.92 & 38.79 & 0.0 & 0.0 & 0.0 \\
5.98 & 11.96 & 44.61 & 35.76 & 11.96 & 31.10 & 0.0 & 0.0 & 0.0 \\
2.0 & 4.0 & 14.92 & 11.96 & 4.0 & 10.40 & 0.0 & 0.0 & 0.0 \\
5.2 & 10.40 & 38.79 & 31.10 & 10.40 & 27.04 & 0.0 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.333 & 0.0 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.333 & 0.0 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.333
\end{pmatrix}.
$$

Figure 1: Calculated optimal dual matrix Y for SDP relaxation of QCQP in our example. See Section 4.4 for details.

Furthermore, we have added another constraint such that the $(1,1)$ element in Y must equal to 1, which becomes as

$$q_5(\boldsymbol{x}) = B_5 \bullet Y = 1,$$

where $B_5 = (b_{ij}^{(5)})$ with $b_{1,1}^{(5)} = 1$.

A SDPA input must be expressed in the primal form (19), thus we have given the input as $F_0 = -A$, $F_{j+1} = -A_j$ for $j = 0, \dots, 5$, $c_1 = \dots = c_4 = 0$ and $c_5 = 1$. As a consequence, we have an SDPA input with only 5 variables and sparse matrices of dimension 9, both of which are smaller than the example in the above.

In this case, SDPA calculation ended after 11 times of iteration with report that it has properly calculated the primal and the dual optimizers. However, the calculated minimum values both for primal and dual optimization was approximately equal to 9.484×10^1, which was larger than the result with the GPGCD method. Furthermore, in the calculated optimal dual matrix Y as shown in Fig. 1, it seems hard to obtain the coefficients in perturbed polynomials because the elements in the matrix do not preserve the structure as shown in (21).

5 Discussions

In this paper, we have considered calculating approximate GCD of univariate polynomials with SDP for solving derived constrained optimization problem. We have tested Lasserre's SDP relaxation and an SDP relaxation for QCQP using an SDP solver SDPA for our test problem.

Although we never obtained satisfactory result in both cases, the test result suggests that, in our approximate GCD problem, formulation by QCQP will be more stable and efficient than using Lasserre's SDP relaxation for SDP solving. It also suggests that will need other ideas for finding more suitable optimizer for the original optimization problem from the calculated result in the relaxed problem.

As we have seen in the example, a relaxed SDP problem may become much larger for small optimization problems. Thus, for more stable and/or efficient computation, we need to reduce the size of original optimization problem and/or make use of various methods (such as in [25]) that reduce the size of SDP relaxation using characteristics of the original optimization problem. Furthermore, we also need a better relaxation method for better estimation of an optimizer for the original optimization problem and an effective method for correcting results calculated in the relaxed problem to find an optimizer for the original optimization problem.

References

[1] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. SIAM, Philadelphia, third edition, 1999.

[2] B. Beckermann and G. Labahn. A fast and numerically stable Euclidean-like algorithm for detecting relatively prime numerical polynomials. *J. Symbolic Comput.*, 26(6):691–714, 1998. Symbolic numeric algebra for polynomials.

[3] P. Chin, R. M. Corless, and G. F. Corliss. Optimization strategies for the approximate GCD problem. In *Proceedings of the 1998 International Symposium on Symbolic and Algebraic Computation*, pages 228–235 (electronic). ACM, 1998.

[4] R. M. Corless, P. M. Gianni, B. M. Trager, and S. M. Watt. The singular value decomposition for polynomial systems. In *Proceedings of the 1995 International Symposium on Symbolic and Algebraic Computation*, pages 195–207. ACM, 1995.

[5] R. M. Corless, S. M. Watt, and L. Zhi. *QR* factoring to compute the GCD of univariate approximate polynomials. *IEEE Trans. Signal Process.*, 52(12):3394–3402, 2004.

[6] I. Z. Emiris, A. Galligo, and H. Lombardi. Certified approximate univariate GCDs. *J. Pure Appl. Algebra*, 117/118:229–251, 1997. Algorithms for algebra (Eindhoven, 1996).

[7] Feng Guo, Mohab Safey El Din, and Lihong Zhi. Global optimization of polynomials using generalized critical values and sums of squares. In *Proceedings of the 2010 International Symposium on Symbolic and Algebraic Computation*, ISSAC '10, pages 107–114, New York, NY, USA, 2010. ACM.

[8] E. Kaltofen, Z. Yang, and L. Zhi. Approximate greatest common divisors of several polynomials with linearly constrained coefficients and singular polynomials. In *Proceedings of the 2006 International Symposium on Symbolic and Algebraic Computation*, pages 169–176, New York, NY, USA, 2006. ACM.

[9] E. Kaltofen, Z. Yang, and L. Zhi. Structured low rank approximation of a Sylvester matrix. In D. Wang and L. Zhi, editors, *Symbolic-Numeric Computation*, Trends in Mathematics, pages 69–83. Birkhäuser, 2007.

[10] Erich Kaltofen, Bin Li, Zhengfeng Yang, and Lihong Zhi. Exact certification of global optimality of approximate factorizations via rationalizing sums-of-squares with floating point scalars. In *Proceedings of the twenty-first international symposium on Symbolic and algebraic computation*, ISSAC '08, pages 155–164, New York, NY, USA, 2008. ACM.

[11] N. K. Karmarkar and Y. N. Lakshman. On approximate GCDs of univariate polynomials. *J. Symbolic Comput.*, 26(6):653–666, 1998. Symbolic numeric algebra for polynomials.

[12] Jean B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM J. Optim.*, 11(3):796–817 (electronic), 2000/01.

[13] Bin Li, Jiawang Nie, and Lihong Zhi. Approximate gcds of polynomials and sparse sos relaxations. *Theor. Comput. Sci.*, 409:200–210, December 2008.

[14] N. Ohsako, H. Sugiura, and T. Torii. A stable extended algorithm for generating polynomial remainder sequence (in Japanese). *Trans. Japan Soc. Indus. Appl. Math*, 7(3):227–255, 1997.

[15] V. Y. Pan. Computation of approximate polynomial GCDs and an extension. *Inform. and Comput.*, 167(2):71–85, 2001.

[16] J. B. Rosen. The gradient projection method for nonlinear programming. II. Nonlinear constraints. *J. Soc. Indust. Appl. Math.*, 9:514–532, 1961.

[17] M. Sanuki and T. Sasaki. Computing approximate GCDs in ill-conditioned cases. In *SNC '07: Proceedings of the 2007 International Workshop on Symbolic-Numeric Computation*, pages 170–179, New York, NY, USA, 2007. ACM.

[18] T. Sasaki and M-T. Noda. Approximate square-free decomposition and root-finding of ill-conditioned algebraic equations. *J. Inform. Process.*, 12(2):159–168, 1989.

[19] A. Schönhage. Quasi-gcd computations. *J. Complexity*, 1(1):118–137, 1985.

[20] K. Tanabe. A geometric method in nonlinear programming. *J. Optim. Theory Appl.*, 30(2):181–210, 1980.

[21] A. Terui. GPGCD, an iterative method for calculating approximate GCD of univariate polynomials, with the complex coefficients. In *Proceedings of the Joint Conference of ASCM 2009 and MACIS 2009*, volume 22 of *COE Lecture Note*, pages 212–221. Faculty of Mathematics, Kyushu University, December 2009.

[22] A. Terui. An iterative method for calculating approximate GCD of univariate polynomials. In *Proceedings of the 2009 International Symposium on Symbolic and Algebraic Computation*, pages 351–358, New York, NY, USA, 2009. ACM Press.

[23] A. Terui. GPGCD, an iterative method for calculating approximate GCD, for multiple univariate polynomials. In V.P. Gerdt, W. Koepf, E.W. Mayr, and E.H. Vorozhtsov, editors, *Computer Algebra in Scientific Computing (Proc. CASC 2010)*, volume 6244 of *Lecture Notes in Computer Science*, pages 238–249. Springer, 2010.

[24] Lieven Vandenberghe and Stephen Boyd. Semidefinite programming. *SIAM Rev.*, 38(1):49–95, 1996.

[25] Hayato Waki, Sunyoung Kim, Masakazu Kojima, and Masakazu Muramatsu. Sums of squares and semidefinite program relaxations for polynomial optimization problems with structured sparsity. *SIAM Journal on Optimization*, 17(1):218–242, 2006.

[26] Makoto Yamashita, Katsuki Fujisawa, Mituhiro Fukuda, Kazuhide Nakata, and Maho Nakata. A high-performance software package for semidefinite programs: SDPA 7. Research Report B-463, Dept. of Mathematical and Computing Science, Tokyo Institute of Technology, Tokyo, Japan, September 2010.

[27] C. J. Zarowski, X. Ma, and F. W. Fairman. QR-factorization method for computing the greatest common divisor of polynomials with inexact coefficients. *IEEE Trans. Signal Process.*, 48(11):3042–3051, 2000.

[28] Z. Zeng. The approximate GCD of inexact polynomials, Part I: a univariate algorithm (extended abstract). preprint, 2004. 8 pages.

[29] Xiao Zheng, Xiao Sun, and Duan Li. Convex relaxations for nonconvex quadratically constrained quadratic programming: matrix cone decomposition and polyhedral approximation. *Mathematical Programming*, 2011.

[30] L. Zhi. Displacement structure in computing approximate GCD of univariate polynomials. In *Computer mathematics: Proc. Six Asian Symposium on Computer Mathematics (ASCM 2003)*, volume 10 of *Lecture Notes Ser. Comput.*, pages 288–298. World Sci. Publ., River Edge, NJ, 2003.