HONORS PROJECT

YIXUAN ZHOU

Abstract. Short description

1. Introduction

Polynomials are very powerful in solving problems in applied math and engineering. And deciding the nonnegativity of a given polynomial become very important in many settings (polynomial optimizations and statistical regressions). Therefore, this decision problem become a central topic in the study of real algebraic geometry. In computational complexity terminology, it has been known that the deciding whether an arbitrary multivariate polynomial is a *nonnegative polynomial* (see Definition 2.5) is a NP-hard problem (computationally infeasible) when the degree of the polynomial is greater than two.

However, if we redirect our focus to determine whether a given polynomial can be expressed in terms of *sum of squares (SOS)* (see Definition 2.6), the problem is reduced to solving a *semidefinite programming (SDP)* (see Definition 2.11) problem which has known algorithms that can solve the problem in polynomial times (efficiently) with respect to the input size. Though Hilbert has showed that the set of all *SOS* coincide with the set of all *nonnegative polynomials* only if the polynomials are univariate, quadratic, or bivariate with degree 4, the *SOS* is still the mostly well-known and well-used certificate in this decision problem due to its simplicity and the existence of efficient solvers. [BPT13]

When it comes to solving the *SDP*, the solver is actually dealing with a set of linear constraints, which can be compactly written in the form $Ax = b$, obtained by the process of *comparison of coefficients (COC)*. Though this process will be formally introduced in section 2.3, the intuition behind it is quite simple. It can be understood in terms of vectors after realizing that the set of n-variate real polynomials with degree less than or equals to d, denoted by $\mathbb{R}[\mathbf{x}]_{n,d}$ form a vector space. Then any polynomial $p(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]_{n,d}$ can be uniquely identified as $p(\mathbf{x}) = \sum_{j=0}^{k} c_j b_j$ given a basis $\mathcal{B} = \{b_0, ..., b_k\}$ of $\mathbb{R}[\mathbf{x}]_{n,d}$. Therefore, the process of *COC* is just formulating the equality constraints by identifying the constants $c_0, ..., c_k$.

Because the linear system that formed by *COC* depends on the basis $\mathcal{B}$, the stability of the linear system, measured by the *condition number* (see Definition 2.13), is drastically fluctuated depends on the $\mathcal{B}$ that we chose. And it has been an ongoing problem of what basis will generate the best result. In this paper, we am going to run experiments, with the computer's aid, of different choices of the basis $\mathcal{B}$ to determine, in different scenarios,

Key words and phrases. some keywords go here.

what is the best choice of basis to carry out the process of *COC* so that the resulted linear system is most stabled.

The paper is organized in the following: In Section 2, we will introduce all the preliminary materials, including the tools that we need through this paper, and the algorithm that will be employed to carry out the *COC*. A brief survey of polynomial bases that are considered in this paper will also be included in this section. In Section 3, the main numerical results of the *condition number* will be presented. And in Section 4, we will present some discussion about the result that is obtained in Section 3 and some thoughts on what are the further efforts that can be made to this problem.

Acknowledgements

I would like to express my deepest gratefulness to my program advisor Mareike Dressler for offering this opportunity to do this honor project. During the past three quarters, she not only has been providing insightful ideas about the direction of this project should be going, but also has been guiding me through the process of doing research. The project could not have done without her support and help. Thank you.

2. Preliminaries

In this section, we are going to introduce all the background materials that will be used to understand the problem and describe what algorithm will be used to carry out the *comparison of coefficients (COC)* process. All the notations will also be introduced in this section, and a brief survey of the polynomial base that will be considered in this paper will be included in the end.

2.1. **Real Coefficient Polynomials.**
First, we would rigorously define the decision problem that is considered in this paper. Thus, we will define the terms that are used related to *polynomials* in this subsection.

Throughout this paper, we will use bold letters for vectors, e.g. $\mathbf{x} = (x_1, ..., x_n) \in \mathbb{R}^n$. The set of all m by n real matrices will be denoated as $\mathbb{R}^{m\times n}$ The ring (set) of real-valued n-variate polynomials will be denoted as $\mathbb{R}[\mathbf{x}]$, and the set of all n-variate real polynomials with degree less than or equals to d will be denoted as $\mathbb{R}[\mathbf{x}]_{n,d}$. Some times, when the variables that we are working with is clear from the context, $\mathbb{R}[\mathbf{x}]_{n,d}$ will be abbreviated as $\mathbb{R}_{n,d}$.

Example 2.1. In mathematics, a polynomial is an expression consisting of variables and coefficients, that involves only the operations of addition, subtraction, multiplication, and non-negative integer exponentiation of variables. Therefore, a polynomial will take the form, $p(x, y, z) = x^4 + 2xyz - 6y + 7$ with x, y, z being variables. This polynomial has 3 variables and has degree 3, thus it belongs to $\mathbb{R}[\mathbf{x}]_{3,4}$.

A quick result that we can get about the set $\mathbb{R}[\mathbf{x}]_{n,d}$ is:

Proposition 2.2. *$\mathbb{R}[\mathbf{x}]_{n,d}$ form a vector space, and a basis of it will be denoted as $\mathcal{B}_{n,d}$.*

Proof. Suppose $p(\mathbf{x}), q(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]_{n,d}$, $c \in \mathbb{R}$, then we have, Then we have $cp \in \mathbb{R}[\mathbf{x}]_{n,d}$ because multiplication by a scalar does not increase the degree nor introduce new variables. And $p + q \in \mathbb{R}[\mathbf{x}]_{n,d}$ for the same reason.

□

A canonical basis to this vector space is the monomial basis

$$\mathcal{B}_{n,d} = \{1, x_1, x_2, ..., x_n, x_1^2, x_1x_2, ..., x_n^d\}$$

And by counting, there are $\binom{n+d}{d} = \frac{n!}{(n-d)!d!}$ elements in the basis of $\mathbb{R}[\mathbf{x}]_{n,d}$.

Remark 2.3. *Given $\mathcal{B}_{n,d}$ be a basis of $\mathbb{R}[\mathbf{x}]_{n,d}$. If we list the elements of $\mathcal{B}_{n,d}$ in a column vector, write as b, then bb^T form a matrix whose upper triangle entries can be collected to form a basis of $\mathbb{R}[\mathbf{x}]_{n,2d}$.*

Example 2.4. Let $\mathcal{B}_{2,1} = x, y, 1$, be a basis of $\mathbb{R}[\mathbf{x}]_{2,1}$, then

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \begin{bmatrix} x & y & 1 \end{bmatrix} = \begin{bmatrix} x^2 & xy & x \\ xy & y^2 & y \\ x & y & 1 \end{bmatrix}$$

Then the upper triangle of the result form a basis of $\mathbb{R}[\mathbf{x}]_{2,2}$.

Then, we will define the terminologies related to the decision problem.

Definition 2.5. Let $P_{n,2d}$ denote the set of nonnegative polynomials with n variables and degree at most $2d$, that is

$$P_{n,2d} = \{p \in \mathbb{R}[\mathbf{x}]_{n,2d} : p(\mathbf{x}) \geq 0, \text{ for all } \mathbf{x} \in \mathbb{R}^d\}$$

The reason that we chose to write $2d$ in the definition is that it only make sense to consider polynomials with even degrees when dealing with both decision problems, because a polynomial with odd degree will be negative when we fix all the other variables and move one variable to positive or negative infinity.

As discussed in the introduction, the decision problem of whether an arbitrary polynomial $p(\mathbf{x}) \in P_{n,2d}$ is impossible to solve it computationally efficiently. However, if we consider the following subset of $P_{n,2d}$, there are efficient algorithms.

Definition 2.6. Let $\Sigma_{n,2d}$ denote the set of polynomials with n variables and degree at most $2d$ that are *Sum of Squares*, that is

$$\Sigma_{n,2d} = \{p \in \mathbb{R}[x]_{n,2d} : \text{ exists } q_1(x), ..., q_k(x) \in \mathbb{R}[x]_{n,d} \ s.t.\ p(x) = \sum_{i=1}^{k} q_i^2(x)\}$$

Notice that $\Sigma_{n,2d} \subset P_{n,2d}$ because sum of squares of real numbers are always going to be nonnegative.

Therefore, the *decision problem* that will be considered in this paper is the following:

$$\text{Given } p(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]_{n,2d}, \text{ decide whether } p(\mathbf{x}) \in \Sigma_{n,2d} \tag{2.1}$$

2.2. **Linear Algebra And Semidefinite Programming.** As we have seen that the $\mathbb{R}[\mathbf{x}]_{n,2d}$ form a vector space, then the tools from linear algebra will play an important role in the analysis. Also, both *SDP* and the measurement of stability of system will require some tools from linear algebra. Thus, we will devote this subsection introducing all the required tools.

In order to properly define the *SDP* and measure the stability of a linear system, we first introduce some tools from linear algebra.

Definition 2.7. Given a matrix $A \in \mathbb{R}^{n\times n}$, we say it is *symmetric* if $A^T = A$. We denote the set of *symmetric matrix* as S^n.

Here we present a famous result of *symmetric matrix.*

Theorem 2.8 ([Gol96])**.** *Spectral Theorem*

Given a symmetric matrix$A \in \mathbb{R}^{n\times n}$*, it can be diagonalized as*

$$A = P^{-1}DP$$

where D is a diagonal matrix with all real values, and P is an orthonormal matrix.

In other word, A has all real eigenvalues, and their corresponding eigenvectors form an orthonormal basis of $\mathbb{R}^n$*.*

Then we introduce the key idea that related to *SDP*, the *positive semidefinite matrix.*

Definition 2.9. Given a matrix $A \in \mathbb{R}^{n\times n}$, it is *positive semidefinite (psd)* if A is symmetric and

$$x^T Ax \geq 0 \quad \forall x \in \mathbb{R}^n$$

And we denote it as $A \succcurlyeq 0$.

Proposition 2.10. *A matrix is psd if and only if all its eigenvalues are greater than or equals to 0*

Proof. If a matrix A has eigenvalue $\lambda < 0$, then if x is the corresponding eigenvector, we have $x^T Ax = \lambda x^T x < 0$. On the other hand, if A has all positive eigenvalues, because A being a symmetric matrix, its eigenvectors form a basis. Thus, for any $x \in \mathbb{R}^n$, we have $x = \sum_{i=1}^n c_i v_i$ where v_i are the eigenvectors of A, that are also orthonormal to each other. Hence, $x^T Ax = \sum_{i=1}^n \lambda_i$. Since all $\lambda_i \geq 0$, we have that $x^T Ax \geq 0$. □

Definition 2.11. A *Semidefinite Problem (SDP)* in standard primal form is written as

$$\textit{minimize } \langle C, X\rangle \quad \textit{subject to } \langle A_i, X\rangle = b_i, i = 1, ..., k \quad X \succcurlyeq 0 \tag{2.2}$$

One can compactly write the constraint $\langle A_i, X\rangle = b_i$ compactly in a matrix form, we can collect all the constraints and write it is $\langle A, X\rangle = \mathbf{b}$ The stability of this linear system is then of the interest of this paper. The *condition number* of the matrix A will be used to measure the stability of the above linear system. The *condition number* can be nicely calculated using the inverse (when the matrix is square) and the *pesudo-inverse* (when the matrix is rectangle) of the matrix A.

Definition 2.12. Given a matrix $A \in \mathbb{R}^{m\times n}$, the *pesudo-inverse*, which is also knows as the *Moore-Penrose* inverse of A, is the matrix $A^\dagger$ satisfying:

- $AA^\dagger A = A$
- $A^\dagger AA^\dagger = A^\dagger$
- $(AA^\dagger)^T = AA^\dagger$
- $(A^\dagger A)^T = AA^\dagger$

Every matrix has its pesudo-inverse, and when $A \in \mathbb{R}^{m\times n}$ is *full rank*, that is $rank(A) = min\{n, m\}$, A can be expressed in simple algebraic form.

In particular, when A has linearly independent columns, $A^\dagger$ can be computed as

$$A^\dagger = (A^T A)^{-1} A^T$$

In this case, the pesudo-inverse is called the *left inverse* since $A^\dagger A = I$.

And when A has linearly independent rows, $A^\dagger$ can be computed as

$$A^\dagger = A^T (AA^T)^{-1}$$

In this case, the pesudo-inverse is called the *right inverse* since $AA^\dagger = I$.

Definition 2.13. Given a matrix $A \in \mathbb{R}^{m\times n}$, the condition number of A, $\kappa(A)$ is defined as

$$\kappa(A) = \begin{cases} ||A|| \cdot ||A^\dagger|| & \textit{if } A \textit{ is full rank} \\ \infty & \textit{otherwise} \end{cases}$$

for any norm $||\cdot||$ imposed on A, for instance, *Frobenius norm.*

Here, I will give a brief description of how the *condition number* is related to the stability of the system by introducing another way to define it.

$$\kappa(A) = \frac{\sigma_{max}(A)}{\sigma_{min}(A)}$$

where the σ denotes the singular values of A.

Thus, it can be understood as how stale our system is. Intuitively, when the condition number is large, some error in the input along the max direction of the singular value, our result would largely fluctuate because the error, magnified by the singular value, will dominate the input that is along the direction of the minimum singular value. Therefore, the smaller the condition number is, the more stable our system is under fluctuations caused by noises. The rigorous explanation of the condition number can be found in [CK07].

2.3. **Comparing Coefficient Algorithm.** With all the tools in hand, we are now ready to introduce the *COC* algorithm that will be used to solve the decision problem described in (2.1).

The algorithm is build upon the following theorem, one can find a detailed explanation of it in the third chapter of [BPT13].

Theorem 2.14. *Given $p(x) \in P_{n,2d}$, if $p(x) \in \Sigma_{n,2d}$, then for any basis $\mathcal{B}_{n,d}$ of $\mathbb{R}_{n,d}$, there exists a matrix such that*

$$\mathcal{B}_{n,d}^T \mathcal{Q} \mathcal{B}_{n,d} = p(x) \text{ and } \mathcal{Q} \succcurlyeq 0 \tag{2.3}$$

Proof. For any $p(x) \in \mathbb{R}_{n,2d}$, if $p(x) \in \Sigma_{n,2d}$, then we can write

$$p(x) = \sum_{i=1}^{k} q^2(x) = \begin{bmatrix} q_1(x), ..., q_k(x) \end{bmatrix} \begin{bmatrix} q_1 \\ \vdots \\ q_k \end{bmatrix}$$

Notice that $q_j(x) \in P_{n,d}$.

Now given $\mathcal{B}_{n,d} = \{b_1, ..., b_{\binom{n+d}{d}}\}$ be a basis of $P_{n,d}$, we have

$$q_j(x) = \sum_{i=1}^{\binom{n+d}{d}} c_j b_j = \begin{bmatrix} c_1, ..., c_{\binom{n+d}{d}} \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ b_{\binom{n+d}{d}} \end{bmatrix}$$

By substituting the section equation into the first, we have

$$p(x) = \begin{bmatrix} b_1 & \cdots & b_{\binom{n+d}{d}} \end{bmatrix} \begin{bmatrix} c_{1,1} & \cdots & c_{1,k} \\ \vdots & & \\ c_{\binom{n+d}{d},1} & \cdots & c_{\binom{n+d}{d},k} \end{bmatrix} \begin{bmatrix} c_{1,1} & \cdots & c_{1,\binom{n+d}{d}} \\ \vdots & & \\ c_{k,1} & \cdots & c_{k,\binom{n+d}{d}} \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ b_{\binom{n+d}{d}} \end{bmatrix}$$

Now the matrices in the middle is $C^T C = \mathcal{Q}$ a *psd* matrix, which proofs the forward direction of this theorem.

On the other hand, if we know $p(x) = \mathcal{B}_{n,d}^T \mathcal{Q} \mathcal{B}_{n,d}$ where $\mathcal{Q}$ is a *psd* matrix, we can just apply the Cholesky decomposition to get $\mathcal{Q} = L^T L$, and recover the SOS form of $p(x)$ as $\mathcal{B}_{n,d}^T L^T L \mathcal{B}_{n,d}$. □

Therefore, we have reduced our problem decision problem to finding this *psd* matrix. The above theorem provides a hint that the above problem can be solved via *SDP*. Actually, it would be solving a sub-part of *SDP*. When examine the formulation of *SDP* in (2.2), we would minimize a target function subject to a set of linear constraints and the variable being a *psd* matrix. By examining (2.3), we can see that we have a set of constraints (later we will see that the set of constraints can be translated exactly into the constraints in *SDP*) and the requirement of $\mathcal{Q}$ being a *psd*. Therefore, we found that the existence condition that is provided in the Theorem 2.14 is a subpart of the *SDP*, which is determining whether there is a feasible point that satisfies the constraints that are imposed.

To address how the constraints $\mathcal{B}_{n,d}^T \mathcal{Q} \mathcal{B}_{n,d} = p(x)$ are translated into the constraints $\langle A, X \rangle = B$ in the *SDP*, we have the following proposition.

Proposition 2.15. *We pick a basis of* $\mathcal{B}_{n,d} = \{b_1, ..., b_{\binom{n+d}{d}}\}$ *of* $\mathbb{R}_{n,d}$*, and list it in a vector form* $\mathbf{b} = \begin{bmatrix} b_1 & \cdots & b_{\binom{n+d}{d}} \end{bmatrix}^T$*. Then by Remark 2.3, we can form a basis* $\mathcal{B}_{n,2d} = \{b'_1, ..., b'_{\binom{n+2d}{2d}}\}$ *from the vector* b*. Suppose the* $p(x)$ *that we are interested in is written in the form* $\sum_{i=1}^{\binom{n+2d}{2d}} c_i b'_i$*. Then, we have the reformulation of the constraints as* $p(x) = \mathbf{b}^T \mathcal{Q} \mathbf{b} = \langle Q, \mathbf{b}\mathbf{b}^T \rangle$*, which when written separately in different rows is exactly the formulation in Definition 2.11.*

Notice that the constraints $p(x) = \langle Q, \mathbf{b}\mathbf{b}^T \rangle$ involved in comparing two polynomials. By theorem 2.2, $\mathbb{R}[\mathbf{x}]_{n,2d}$ form a vector space. The equality of two vectors is established by comparing the coordinates of the two vectors when under the same basis. Thus, when the basis of $p(x)$ is the same as the basis formed by the upper triangle of $\mathbf{b}\mathbf{b}^T$, we can compare the coordinates of the vectors to establish the equality. And the coordinates for polynomials are called their coefficients. Thus, we name this process as *Comparing of Coefficients (COC)* and the paper is designated to evaluate the stability of the constrants $p(x) = \langle Q, \mathbf{b}\mathbf{b}^T \rangle$.

As I have mentioned, the stability will be measured by the *condition number* of a matrix. Thus, we would need to re-write the constraints in to the form $A\S = \rfloor$. Therefore, the problem need to be further reformulated. And how the choices of the basis are involved in this process will also be introduced.

Definition 2.16 ([Rec14])**.** We call the matrix bb^T in Proposition 2.15 the *Moment Matrix*, we will denote this matrix as $\mathcal{M}$, and it is a symmetric matrix by definition.

Example 2.17. Here is another place to do an example to illustrate Moment Matrix.

The *Moment Matrix* provides the first choice of basis that is involved in the process of *COC*. Suppose the given polynomial $p(x) = \sum_{j=0}^{k} c_j b_j$ is in the same basis as the resulted basis of the *Moment Matrix*, that is the upper triangle of $\mathbf{bb^T}$ consists $b_0, ..., b_k$. Because $\mathcal{Q} \succcurlyeq 0$, $\mathcal{Q}$ is symmetric, it is completely determined by its upper triangle. Let $\mathbf{q} = \left[q_{0,0}, ..., q_{0,m}, q_{1,1}, q_{1,2}, ..., q_{m,m}\right]^T$ be the vector consists of all the elements of the upper triangle of $\mathcal{Q}$. The constrants $p(x) = \langle Q, \mathbf{b}\mathbf{b}^T\rangle$ can then be reformulated as a set of linear equations $A\mathbf{q} = \mathbf{c}$ where $\mathbf{c} = \left[c_0, ..., c_k\right]^T$ is the vector of the coefficients of the $p(x)$, and A is a matrix that is used to establish the equality of polynomials. Then, we can measure the stability of the constraints $p(x) = \langle Q, \mathbf{b}\mathbf{b}^T\rangle$ by the *condition number* of A.

Since, this matrix A is not going to be the final matrix that is going to be used in analysis, the procedure of obtaining A will be omitted.

Now, what if the $p(x)$ is written in a basis that is different from the basis constructed by the *Moment Matrix*? One might argue that we can simply apply a change of basis matrix to convert $p(x)$ into the basis that is used in *Moment Matrix*. However, that is no efficient and accurate way to obtain a change of basis matrix. The reason is that a change of basis matrix would involve writing the basis of one polynomial in terms of the basis of the other. This itself is a huge process of *Comparing of Coefficients* and would result in an increase of perturbation to the system because the *condition number* is never smaller than 1 [Gol96]. Therefore, we shall introduce the *Coefficient Moment Matrix*. In the remaining part of this section, we shall build our way to it.

Suppose the *Moment Matrix* is constructed with the basis $\mathcal{B}_{n,d}$ and the polynomial $p(x)$ is written in the basis $\mathcal{B}'_{n,2d}$. That is supposed $\mathcal{B}'_{n,2d} = \{b'_0, ..., b'_k\}$ where $k = \binom{n+2d}{2d} - 1$, we have $p(x) = c_0 b'_0 + ... + c_k b'_k$.

Definition 2.18 ([Rec14])**.** We define the *coefficient extraction map* as the following map,

$$\mathcal{C} : \mathbb{R}[x]_{\leq,2d} \times \mathbb{R}[x]_{\leq,2d} \to \mathbb{R}$$
$$\mathcal{C}(p, b'_j) \mapsto c_j$$

When fixing s_j, we have the *coefficient extraction map* being a linear map with respect to the polynomial $p(x)$. Indeed, we have

$$\mathcal{C}(\lambda p, b'_j) = \lambda \mathcal{C}(p, b'_j) \quad \lambda \in \mathbb{R}$$
$$\mathcal{C}(p_1 + p_2, b'_j) = \mathcal{C}(p_1, b'_j) + \mathcal{C}(p_2, b_j)$$

Remark 2.19. *When the $\mathcal{B}'{}_{n,2d} = \{b'_1, ..., b'_k\}$ is an orthonormal basis, i.e. $\langle b_i, b_j\rangle\delta_{i,j}$ (the Dirac delta function), where the inner product is defined as*

$$\langle p, q\rangle = \int_a^b p(x)q(x)d\alpha(x)$$

There is a natural concretely definition for the coefficients extraction map*. That is*

$$\mathcal{C}(p(x), b'_j) = \langle p(x), b'_j\rangle$$

When the basis is only orthogonal, we can still define the coefficients extraction map *concretely as,*

$$\mathcal{C}(p(x), b'_j) = \frac{1}{||b'_j||}\langle p(x), b'_j\rangle$$

where the norm $||\cdot||$ is induced by the corresponding inner product.

Remark 2.20. *We should actually write the* coefficient extraction map *as* $\mathcal{C}_{\mathcal{B}'_{n,2d}}$*, since it depends on the base itself. However, when the base is clear, we just write it as* $\mathcal{C}$*.*

We can generalize the *coefficients extraction map* to take in a matrix as the first argument, and just entry-wise apply the map. With an abuse of notation, we have

Definition 2.21. Given a matrix of polynomials $(p_{i,j}(x))_{i,j}$ all in the bases Let the *coefficient extraction map* be defined as

$$\mathcal{C} : \mathbb{R}[x]^{m\times n}_{\leq n,2d} \times \mathbb{R}[x]_{\leq n,2d} \to \mathbb{R}^{m\times n}$$

$$\mathcal{C}(\begin{bmatrix} p_{1,1} & \dots & p_{1,n} \\ & \vdots & \\ p_{m,1} & \dots & p_{m,n} \end{bmatrix}, b'_j) = \begin{bmatrix} \mathcal{C}(p_{1,1}, b'_j) & \dots & \mathcal{C}(p_{1,n}, b'_j) \\ & \vdots & \\ \mathcal{C}(p_{m,1}, b'_j) & \dots & \mathcal{C}(p_{m,n}, b'_j) \end{bmatrix}$$

Remark 2.22. *An immediate result from the above definition is that, given* $Q \in \mathbb{R}^{m\times m}$*,* $M \in \mathbb{R}[x]^{m\times m}_{n,2d}$*, and a basis* $\mathcal{B}'_{n,2d} = \{b'_1, ..., b'_k\}$*, the matrix inner product provides the following relation,*

$$\mathcal{C}(\langle Q, M\rangle, b'_j) = \langle Q, \mathcal{C}(M, b'_j)\rangle$$

Notice that, let $\mathcal{B}_{n,d} = \{b_0, ..., b_l\}$, where $l = \binom{n+d}{d} - 1$, let $\mathbf{b} = [b_1, ..., b_l]^T$, $M = \mathbf{b}\cdot\mathbf{b}^T \in \mathbb{R}^{l,l}$. Then given $p(x)$ in $\mathcal{B}'_{n,2d} = \{b'_0, ..., b'_k\}$, $p = c_0 b'_0 + ... + c_k b'_k$, given $Q \in \mathbb{R}^{l,l}$ be the change of basis matrix from $\mathcal{B}_{n,2d}$ to $\mathcal{B}'_{n,2d}$, where $\mathcal{B}_{n,2d}$ is generated by $\mathcal{B}_{n,d}$ using remark 2.3, we have

$$c_j = \mathcal{C}(p, b'_j) = \mathcal{C}(\langle Q, M\rangle, b'_j) = \langle Q, \mathcal{C}(M, b'_j)\rangle \tag{2.4}$$

Definition 2.23. Define the matrix $\mathcal{A}_j = \mathcal{C}(M, b'_j)$ be the *coefficient moment matrix* of b'_j.

Proposition 2.24. $\mathcal{A}_j$ *is symmetric, because* M *is symmetric.*

Recall, the *SOS* problem is to decide, given a polynomial $p(x)$, whether there exists a $Q \succcurlyeq 0$ such that $p = \mathbf{b}^T Q\mathbf{b}$, where $\mathbf{b}$ be the vector generated by $\mathcal{B}_{n,d}$. Suppose $p(x)$ is given in $\mathcal{B}'_{n,2d}$, we can then reformulate the constraints $\mathbf{b}^T Q\mathbf{b}$ using the *coefficient moment matrix* as

$$\langle Q, \mathcal{A}_j\rangle = c_j \quad \forall j = 0, ..., \binom{n+2d}{2d} - 1 \tag{2.5}$$

Let $\mathcal{A}_j = \begin{bmatrix} a_{0,0} & a_{0,1} & \dots & a_{0,l} \\ a_{0,1} & a_{1,1} & \dots & a_{1,l} \\ & \vdots & & \\ a_{0,l} & a_{1,l} & \dots & a_{l,l} \end{bmatrix}, Q = \begin{bmatrix} q_{0,0} & q_{0,1} & \dots & q_{0,l} \\ q_{0,1} & q_{1,1} & \dots & q_{1,l} \\ & \vdots & & \\ q_{0,l} & q_{1,l} & \dots & q_{l,l} \end{bmatrix}$

Set $\mathbf{a_j} = [a_{0,0}, 2a_{0,1} ..., 2a_{0,l}, a_{1,1}, 2a_{1,2}, ..., a_{l,l}]^T \in \mathbb{R}^{l(l+1)/2}$, and $\mathbf{q} = [q_{0,0}, ..., q_{1,1}, q_{1,2} ..., q_{l,l}]^T \in \mathbb{R}^{l(l+1)/2}$. We can re-write the inner product $\langle Q, \mathcal{A}_j\rangle$ using the fact that both $\mathcal{A}_j$ and Q are symmetric.

$$\langle Q, \mathcal{A}_j\rangle = \mathbf{q}^T \cdot \mathbf{a_j} = \mathbf{a_j}^T \cdot \mathbf{q}$$

Then, finally, we can re-write the constraint $p(x) = \mathbf{b}^T Q\mathbf{b}$, as the system of linear equations that

$$\mathcal{A}\mathbf{q} = \begin{bmatrix} a_0^T \\ \vdots \\ a_l^T \end{bmatrix} \mathbf{q} = \begin{bmatrix} c_0 \\ \vdots \\ c_l \end{bmatrix} = \mathbf{c}$$

Thus, the numerical property of the *SDP* problem is completely captured by the *condition number* of A. Therefore, we will write python code to examine different combinations of bases under different degrees and number of variates in the Preliminaries sections. We will list the polynomial bases that we are interested in the following sections, and will briefly touch upon *Semidifinite Programing* before we present our results.

2.4. **Polynomial Basis.**

2.5. **Solving Semidefinite Program.** Toy examples maybe

3. Numerical Results

Proposition 3.1.

Proof. □

maybe a theorem

Theorem 3.2.

or an example...

Example 3.3.

pictures are always a good idea...

4. Maybe Some Proofs

5. Resume, Outlook, or/and Open Problems

what did you do, what questions are still open, natural next steps etc.

References

[BPT13] G. Blekherman, P.A. Parrilo, and R.R. Thomas, *Semidefinite optimization and convex algebraic geometry*, MOS-SIAM Series on Optimization, vol. 13, SIAM and the Mathematical Optimization Society, Philadelphia, 2013.

[CK07] E. Cheney and D. Kincaid, *Numerical mathematics and computing*, International student edition, Cengage Learning, 2007.

[Gol96] Gene Golub, *Matrix computations*, Johns Hopkins University Press, Baltimore, 1996.

[Rec14] M. Recher, *Zur numerischen Auswirkung von Basiswahlen in der polynomiellen Optimierung*, 2014, Master Thesis, Universität zu Köln.

8514 Villa La Jolla Drive # 114, La Jolla, CA, 92037

Email address: yiz044@ucsd.edu