# EXPLOITING SPARSITY IN SOS PROGRAMMING AND SPARSE POLYNOMIAL OPTIMIZATIONS

JIE WANG

Abstract. In this paper, we consider a new pattern of sparsity for SOS Programming named by cross sparsity patterns. We use matrix decompositions for a class of PSD matrices with chordal sparsity patterns to construct sets of supports for a sparse SOS decomposition. The method is applied to the certificate of the nonnegativity of sparse polynomials and unconstrained sparse polynomial optimization problems. Many numerical experiments are given. It turns out that our method can dramatically reduce the computational cost and can handle really huge polynomials, for example, polynomials with 10 variables, of degree 40 and more than 5000 terms.

## 1. Introduction

Certificates of nonnegative polynomials and polynomial optimization problems (POPs) arise from many fields such as mathematics, control, engineering, statistics and physics. A classical method for these problems is using sums of squares (SOS) programming which can be effectively solved by semidefinite program (SDP) ([23, 24]). However, when the given polynomial has many variables and a high degree, the size of corresponding matrices for SDP is very large and the existing SDP solvers are hard to deal with. On the other hand, most polynomials coming from practice have certain structures including symmetry and sparsity. So it is very important to take full advantage of structures of polynomials to reduce the size of corresponding SDP problems. A lot of work has been done on this subject ([4, 6, 7, 11, 14, 15, 16, 18, 22, 25, 26, 27, 29, 30, 31, 32, 33, 35]).

For a polynomial $f \in \mathbb{R}[\mathbf{x}] = \mathbb{R}[x_1, \ldots, x_n]$, if we choose a monomial basis $M = \{\mathbf{x}^{\boldsymbol{\omega}_1}, \ldots, \mathbf{x}^{\boldsymbol{\omega}_r}\}$, then the SOS condition can be converted to the problem of deciding if there exists a positive semidefinite matrix $Q$ (Gram matrix) such that $f(x) = M^T Q M$. There are two approaches to reduce computations. One approach is reducing the size of the monomial basis $M$; such techniques include using Newton polytopes ([27]), the diagonal inconsistency ([18]), the iterative elimination method ([16]), and the facial reduction ([25, 30, 31]). The second approach is exploiting the sparsity of the Gram matrix $Q$; such techniques include the correlative sparsity ([6, 22, 29, 33]), the symmetry property ([11]), the split property ([7]), minimal coordinate projections ([26]), and the coefficient matching conditions ([4, 14, 35]).

In this paper, we consider a new pattern of sparsity for SOS Programming named by *cross sparsity patterns*. Given a polynomial $f$ with the support set $\mathscr{A} \subseteq \mathbb{N}^n$, the

*Date*: September 19, 2018.

2010 *Mathematics Subject Classification.* Primary, 14P10,90C25; Secondary, 52B20,12D15.

*Key words and phrases.* nonnegative polynomial, sparse polynomial, polynomial optimization, sum of squares, chordal graph.

This work was supported partly by NSFC under grants 61732001 and 61532019.

cross sparsity pattern associated with $\mathscr{A}$ is described in terms of a $r \times r$ symmetric $(0,1)$-matrix $R_{\mathscr{A}}$ whose elements are given by

$$R_{ij} = \begin{cases} 1, & \boldsymbol{\omega}_i + \boldsymbol{\omega}_j \in (2\mathbb{N})^n \cup \mathscr{A}, \\ 0, & \text{otherwise.} \end{cases} \tag{1.1}$$

From the cross sparsity pattern matrix $R_{\mathscr{A}}$, we associate it with a undirected graph $G(V_{\mathscr{A}}, E_{\mathscr{A}})$ with $V_{\mathscr{A}} = \{1, 2, \ldots, r\}$ and $E_{\mathscr{A}} = \{\{i, j\} \mid i, j \in V_{\mathscr{A}}, i < j, R_{ij} = 1\}$. The key idea in this paper is to use matrix decompositions for a class of positive semidefinite matrices with chordal sparsity patterns to construct sets of supports for a sparse SOS decomposition. Concretely, suppose that $\tilde{G}(V_{\mathscr{A}}, \tilde{E}_{\mathscr{A}})$ is a chordal extension of $G(V_{\mathscr{A}}, E_{\mathscr{A}})$, and $C_1, C_2, \ldots, C_t \subseteq V_{\mathscr{A}}$ denote the maximal cliques of $\tilde{G}(V_{\mathscr{A}}, \tilde{E}_{\mathscr{A}})$. Let $\mathscr{B}_k = \{\boldsymbol{\omega}_i \mid i \in C_k\}$ for $i = 1, 2, \ldots, t$. Then we take $f = \sum_{k=1}^t f_k^2$ as a sparse SOS relaxation for the nonnegativity of the sparse polynomial $f$, where $f_k$ has the support set $\mathscr{B}_k$ for $k = 1, \ldots, t$. If the size of the cliques $C_k, k = 1, \ldots, t$ is small, this can reduce the computational cost. This is somewhat similar to the use of correlative sparsity patterns in [29]. However, correlative sparsity patterns focus on the sparsity of variables and our cross sparsity patterns are more general.

We test our method on many examples. It turns out that our method can dramatically reduce the computational cost and can handle really huge polynomials, for example, polynomials with 10 variables, of degree 40 and more than 5000 terms.

The rest of the paper organized as follows. In section 2, we introduce some basic notions from nonnegative polynomials and graph theory. In section 3, we define a cross sparsity pattern associated with a sparse polynomial. We show that how we can exploit this sparsity pattern to obtain a sparse SOS relaxation for the nonnegativity of the sparse polynomial. In section 4, we apply this sparse SOS relaxation to unconstrained sparse POPs. We discuss in section 5 when the sparse SOS relaxation obtains the same optimal values as the dense SOS relaxation. Section 6 includes numerical results on various examples. We show that the proposed sparse SOS relaxation exhibits significantly better performance in practice. Finally, we give conclusions in section 7.

## 2. Preliminaries

2.1. **Nonnegative Polynomials.** Let $\mathbb{R}[\mathbf{x}] = \mathbb{R}[x_1, \ldots, x_n]$ be the ring of real $n$-variate polynomial. For a finite set $\mathscr{A} \subset \mathbb{N}^n$, we denote by $\mathrm{conv}(\mathscr{A})$ the convex hull of $\mathscr{A}$, and by $V(\mathscr{A})$ the vertices of the convex hull of $\mathscr{A}$. Also we denote by $V(P)$ the vertex set of a polytope $P$. A polynomial $f \in \mathbb{R}[\mathbf{x}]$ can be written as $f(\mathbf{x}) = \sum_{\boldsymbol{\alpha} \in \mathscr{A}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$ with $c_{\boldsymbol{\alpha}} \in \mathbb{R}, \mathbf{x}^{\boldsymbol{\alpha}} = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$. The support of $f$ is defined by $\mathrm{supp}(f) = \{\boldsymbol{\alpha} \in \mathscr{A} \mid c_{\boldsymbol{\alpha}} \neq 0\}$, the degree of $f$ is defined by $\deg(f) = \max\{\sum_{i=1}^n \alpha_i : \boldsymbol{\alpha} \in \mathrm{supp}(f)\}$, and the Newton polytope of $f$ is defined as $\mathrm{New}(f) = \mathrm{conv}(\{\boldsymbol{\alpha} : \boldsymbol{\alpha} \in \mathrm{supp}(f)\})$.

A polynomial $f \in \mathbb{R}[\mathbf{x}]$ which is nonnegative over $\mathbb{R}^n$ is called a *nonnegative polynomial*. The class of nonnegative polynomials is denoted by PSD, which is a convex cone.

A vector $\boldsymbol{\alpha} \in \mathbb{N}^n$ is *even* if $\alpha_i$ is an even number for $i = 1, \ldots, n$. A necessary condition for a polynomial $f(\mathbf{x})$ to be nonnegative is that every vertex of its Newton polytope is an even vector, i.e. $V(\mathrm{New}(f)) = V(\mathrm{supp}(f)) \subseteq (2\mathbb{N})^n$ ([27]).

For a nonempty finite set $\mathscr{B} \subseteq \mathbb{N}^n$, $\mathbb{R}[\mathscr{B}]$ denotes the set of polynomials in $\mathbb{R}[\mathbf{x}]$ whose supports are contained in $\mathscr{B}$, i.e., $\mathbb{R}[\mathscr{B}] = \{f \in \mathbb{R}[\mathbf{x}] \mid \mathrm{supp}(f) \subseteq \mathscr{B}\}$ and we

use $\mathbb{R}[\mathscr{B}]^2$ to denote the set of polynomials which are sums of polynomials in $\mathbb{R}[\mathscr{B}]$. Let $\mathbf{x}^{\mathscr{B}}$ be the $|\mathscr{B}|$-dimensional column vector consisting of elements $\mathbf{x}^{\boldsymbol{\beta}}, \boldsymbol{\beta} \in \mathscr{B}$, then

$$\mathbb{R}[\mathscr{B}]^2 = \{(\mathbf{x}^{\mathscr{B}})^T Q \mathbf{x}^{\mathscr{B}} \mid Q \in S_+^{|\mathscr{B}|}\},$$

where the matrix $Q$ is called the Gram matrix.

2.2. **Chordal Graphs.** A *graph* $G(V, E)$ consists of a set of nodes $V = \{1, 2, \ldots, r\}$ and a set of edges $E \subseteq V \times V$. A graph $G(V, E)$ is said to be *undirected* if and only if $(i, j) \in E \Leftrightarrow (j, i) \in E$. A *cycle* of length $k$ is a sequence of nodes $\{v_1, v_2, \ldots, v_k\} \subseteq V$ with $(v_k, v_1) \in E$ and $(v_i, v_{i+1}) \in E$, for $i = 1, \ldots, k-1$. A *chord* in a cycle $\{v_1, v_2, \ldots, v_k\}$ is an edge $(v_i v_j)$ that joins two nonconsecutive nodes in the cycle.

**Definition 2.1.** *An undirected graph is* chordal *if all its cycles of length at least four have a chord.*

Chordal graphs include some common classes of graphs, such as complete graphs, line graphs and trees. Note that any non-chordal graph $G(V, E)$ can always be extended to a chordal graph $\tilde{G}(V, \tilde{E})$ by adding appropriate edges to $E$. Finally, we introduce the concept of cliques: a *clique* $C \subseteq V$ is a subset of nodes where $(i, j) \in E, \forall i, j \in C, i \neq j$. If a clique $C$ is not a subset of any other clique, then it is called a *maximal clique*. It is known that maximal cliques of a chordal graph can be enumerated efficiently in linear time in the number of vertices and edges of the graph. See [10, 12] for chordal graphs and finding all maximal cliques.

Given an undirected graph $G(V, E)$, we define an extended set of edges $E^\star := E \cup \{(i, i) \mid i \in V\}$ that includes all selfloops. Then, we define the space of symmetric sparse matrices as

$$S^r(E, 0) := \{X \in S^r \mid X_{ij} = X_{ji} = 0 \text{ if } (i, j) \in E^\star\} \tag{2.1}$$

and the cone of sparse PSD matrices as

$$S_+^r(E, 0) := \{X \in S^r(E, 0) \mid X \succeq 0\}. \tag{2.2}$$

Given a maximal clique $C_k$, we define a matrix $P_{C_k} \in \mathbb{R}^{|C_k| \times r}$ as

$$(P_{C_k})_{ij} = \begin{cases} 1, & C_k(i) = j, \\ 0, & \text{otherwise.} \end{cases} \tag{2.3}$$

where $C_k(i)$ denotes the i-th node in $C_k$, sorted in the natural ordering. Note that $X_k = P_{C_k} X_k P_{C_k}^T \in S^{|C_k|}$ extracts a principal submatrix defined by the indices in clique $C_k$, and the operation $E_{C_k}^T X_k E_{C_k}$ inflates a $|C_k| \times |C_k|$ matrix into a sparse $r \times r$ matrix. Then, the following results characterize, respectively, the membership to the set $S_+^r(E, 0)$ when the underlying graph $G(V, E)$ is chordal.

**Theorem 2.2** ([1])**.** *Let $G(V, E)$ be a chordal graph with maximal cliques $\{C_1, \ldots, C_t\}$. Then $X \in S_+^r(E, 0)$ if and only if there exist $X_k \in S_+^{|C_k|}, k = 1, \ldots, t$ such that $X = \sum_{k=1}^t P_{C_k}^T X_k P_{C_k}$.*

## 3. Exploiting sparsity in SOS programming

A basic problem that appears in many fields is checking global nonnegativity of multivariate polynomials. This is difficult in general. A convenient approach for this, originally introduced by Parrilo in [23], is the use of sums of squares as a

suitable replacement for nonnegativity. Given a polynomial $f(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$, if there exist polynomials $f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})$ such that

$$f(\mathbf{x}) = \sum_{i=1}^{m} f_i(\mathbf{x})^2, \tag{3.1}$$

then we say $f(\mathbf{x})$ is a *sum of squares* (SOS). The existence of an SOS decomposition of a given polynomial gives a certificate for its global nonnegativity. For $d \in \mathbb{N}$, let $\mathbb{N}^n_d := \{\boldsymbol{\alpha} \in \mathbb{N}^n \mid \sum_{i=1}^n \alpha_i \leq d\}$ and assume $f \in \mathbb{R}[\mathbb{N}^n_{2d}]$. The SOS condition (3.1) can be converted to the problem of deciding if there exists a positive semidefinite matrix $Q$ such that

$$f(\mathbf{x}) = (\mathbf{x}^{\mathbb{N}^n_d})^T Q \mathbf{x}^{\mathbb{N}^n_d}, \tag{3.2}$$

which can be efficiently solved by a semidefinite programming problem (SDP).

We say that a polynomial $f \in \mathbb{R}[\mathbb{N}^n_{2d}]$ is *sparse* if the number of elements in its support $\mathscr{A} = \mathrm{supp}(f)$ is much smaller than the number of elements in $\mathbb{N}^n_{2d}$ that forms a support of fully dense polynomials in $\mathbb{R}[\mathbb{N}^n_{2d}]$. When $f(\mathbf{x})$ is a sparse polynomial in $\mathbb{R}[\mathbb{N}^n_{2d}]$, the size of the SDP problem (3.2) can be reduced by eliminating redundant elements from $\mathbb{N}^n_d$. In fact, $\mathbb{N}^n_d$ in problem (3.2) can be replaced by ([27])

$$\mathscr{B} = \mathrm{conv}(\{\frac{\boldsymbol{\alpha}}{2} \mid \boldsymbol{\alpha} \in V(\mathscr{A})\}) \cap \mathbb{N}^n \subseteq \mathbb{N}^n_d. \tag{3.3}$$

There are also other methods to reduce the size of $\mathscr{B}$ further ([16, 25, 30]). However, we assume in this paper that $\mathscr{B}$ is as (3.3).

3.1. **Cross Sparsity Pattern.** Let $f(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ with $\mathrm{supp}(f) = \mathscr{A}$. Assume that $\mathscr{B}$ is as (4.4) and $\mathscr{B} = \{\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_r\}$. The sparsity considered in this paper is measured by the different kinds of cross product of monomials arising in the objective polynomial $f(\mathbf{x})$. It is represented by a $r \times r$ *cross sparsity pattern matrix* $\mathbf{R}_{\mathscr{A}} = (R_{ij})$ whose elements are given by

$$R_{ij} = \begin{cases} 1, & \boldsymbol{\omega}_i + \boldsymbol{\omega}_j \in (2\mathbb{N})^n \cup \mathscr{A}, \\ 0, & \text{otherwise.} \end{cases} \tag{3.4}$$

Given a cross sparsity pattern matrix $\mathbf{R}_{\mathscr{A}} = (R_{ij})$, the graph $G(V_{\mathscr{A}}, E_{\mathscr{A}})$ with $V_{\mathscr{A}} = \{1, 2, \ldots, r\}$ and $E_{\mathscr{A}} = \{\{i, j\} \mid i, j \in V_{\mathscr{A}}, i < j, R_{ij} = 1\}$ is called the *cross sparsity pattern graph*. To apply Theorem 2.2, we generate a chordal extension $\tilde{G}(V_{\mathscr{A}}, \tilde{E}_{\mathscr{A}})$ of the cross sparsity pattern graph $G(V_{\mathscr{A}}, E_{\mathscr{A}})$ and use the extended cross sparsity pattern graph $\tilde{G}(V_{\mathscr{A}}, \tilde{E}_{\mathscr{A}})$ instead of $G(V_{\mathscr{A}}, E_{\mathscr{A}})$.

Given a graph $G(V_{\mathscr{A}}, E_{\mathscr{A}})$, there may be many different chordal extensions and choosing anyone of them is valid for deriving the sparse relaxation presented in this paper. For example, we can take all of the connected components of $G(V_{\mathscr{A}}, E_{\mathscr{A}})$ and add edges such that every connected component becomes a complete subgraph. The obtained graph will be a simple choice for chordal extensions. The chordal extension with the least number of edges is called the *minimum chordal extension*. The minimum chordal extension serves as the best choice for the resulting sparse relaxation. However, finding the minimum chordal extension of a graph is an NP-hard problem in general. Finding a chordal extension of a graph is equivalent to calculating the symbolic sparse Cholesky factorization of its adjacency matrix. The resulted sparse matrix represents a chordal extension. The minimum chordal

extension corresponds to the sparse Cholesky factorization with the minimum fill-ins. Fortunately, several heuristic algorithms, such as the minimum degree ordering, are known to efficiently produce a good approximation. For more information on symbolic Cholesky factorizations with the minimum degree ordering and minimum chordal extensions, please refer to [2, 3, 13].

3.2. **Sparse SOS relaxations.** Given $\mathscr{A} \subseteq \mathbb{N}^n$ with $V(\mathscr{A}) \subseteq (2\mathbb{N})^n$, $\mathscr{B}$ is as (3.3). Let the set of SOS polynomials supported on $\mathscr{A}$ be

$$\Sigma(\mathscr{A}) := \{f \in \mathbb{R}[\mathscr{A}] \mid \exists Q \in S^r_+ \text{ s.t. } f = (\mathbf{x}^{\mathscr{B}})^T Q \mathbf{x}^{\mathscr{B}}\}.$$

Generally the Gram matrix $Q$ for a sparse SOS polynomial $f(\mathbf{x})$ can be dense. To maintain the sparsity of $f(\mathbf{x})$ in the Gram matrix $Q$, we consider a subset of SOS polynomials

$$\tilde{\Sigma}(\mathscr{A}) := \{f \in \mathbb{R}[\mathscr{A}] \mid \exists Q \in S^r_+(\tilde{E}_{\mathscr{A}}, 0) \text{ s.t. } f = (\mathbf{x}^{\mathscr{B}})^T Q \mathbf{x}^{\mathscr{B}}\}.$$

With this restriction, we have the following result.

**Theorem 3.1.** *Given $\mathscr{A} \subseteq \mathbb{N}^n$ with $V(\mathscr{A}) \subseteq (2\mathbb{N})^n$, assume that $\mathscr{B} = \{\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_r\}$ is as (3.3) and the cross sparsity pattern graph is $\tilde{G}(V_{\mathscr{A}}, \tilde{E}_{\mathscr{A}})$. Let $C_1, C_2, \ldots, C_t \subseteq V_{\mathscr{A}}$ denote the maximal cliques of $\tilde{G}(V_{\mathscr{A}}, \tilde{E}_{\mathscr{A}})$ and $\mathscr{B}_k = \{\boldsymbol{\omega}_i \in \mathscr{B} \mid i \in C_k\}, i = 1, 2, \ldots, t$. Then, $f(\mathbf{x}) \in \tilde{\Sigma}(\mathscr{A})$ if and only if there exist $f_k(\mathbf{x}) \in \mathbb{R}[\mathscr{B}_k], k = 1, \ldots, t$ such that*

$$f(\mathbf{x}) = \sum_{k=1}^{t} f_k(\mathbf{x})^2. \tag{3.5}$$

*Proof.* By Theorem 2.2, $Q \in S^r_+(\tilde{E}_{\mathscr{A}}, 0)$ if and only if there exist $Q_k \in S^{|C_k|}_+, k = 1, \ldots, t$ such that $Q = \sum_{k=1}^t P_{C_k}^T Q_k P_{C_k}$. So $f(\mathbf{x}) \in \tilde{\Sigma}(\mathscr{A})$ if and only if there exist $Q_k \in S^{|C_k|}_+, k = 1, \ldots, t$ such that

$$\begin{aligned} f(\mathbf{x}) &= (\mathbf{x}^{\mathscr{B}})^T (\sum_{k=1}^{t} P_{C_k}^T Q_k P_{C_k}) \mathbf{x}^{\mathscr{B}} \\ &= \sum_{k=1}^{t} (P_{C_k} \mathbf{x}^{\mathscr{B}})^T Q_k (P_{C_k} \mathbf{x}^{\mathscr{B}}) \\ &= \sum_{k=1}^{t} (\mathbf{x}^{\mathscr{B}_k})^T Q_k \mathbf{x}^{\mathscr{B}_k}, \end{aligned}$$

which is equivalent to that there exist $f_k(\mathbf{x}) \in \mathbb{R}[\mathscr{B}_k], k = 1, \ldots, t$ such that $f(\mathbf{x}) = \sum_{k=1}^t f_k(\mathbf{x})^2$. □

## 4. Sparse polynomial optimization

We consider the unconstrained polynomial optimization problem:

$$\text{minimize} \quad f(\mathbf{x}). \tag{4.1}$$

We first convert the POP (4.1) into an equivalent problem,

$$\begin{cases} \text{maximize} & \xi \\ \text{subject to} & f(\mathbf{x}) - \xi \geq 0. \end{cases} \tag{4.2}$$

Let $\xi^*$ denote the optimal value of (4.2). Assume $f \in \mathbb{R}[\mathbb{N}^n_{2d}]$. Then we can replace the constraint of the problem (4.2) by an SOS constraint to obtain

$$\begin{cases} \text{maximize} & \xi \\ \text{subject to} & f(\mathbf{x}) - \xi \in \mathbb{R}[\mathbb{N}^n_d]^2. \end{cases} \tag{4.3}$$

Let $\xi^*_{sos}$ denote the optimal value of (4.3). The SOS optimization problem (4.3) serves as a relaxation of the POP (4.2). Note that we can rewrite the SOS constraint of (4.3) as $f(\mathbf{x}) - \xi = (\mathbf{x}^{\mathbb{N}^n_d})^T Q \mathbf{x}^{\mathbb{N}^n_d}$ and $Q \in S_+^{|\mathbb{N}^n_d|}$.

When the objective function $f(\mathbf{x})$ is a sparse polynomial in $\mathbb{R}[\mathbb{N}^n_{2d}]$, the SOS constraint of (4.3) can be replaced by $f(\mathbf{x}) - \xi \in \mathbb{R}[\mathscr{B}]^2$ with

$$\mathscr{B} = \text{conv}(\{\frac{\boldsymbol{\alpha}}{2} \mid \boldsymbol{\alpha} \in \text{supp}(f)\} \cup \{\mathbf{0}\}) \cap \mathbb{N}^n \subseteq \mathbb{N}^n_d. \tag{4.4}$$

Note that $\mathbf{0}$ is added as the support for the real number variable $\xi$. Therefore, we obtain

$$\begin{cases} \text{maximize} & \xi \\ \text{subject to} & f(\mathbf{x}) - \xi \in \mathbb{R}[\mathscr{B}]^2. \end{cases} \tag{4.5}$$

Let $\mathscr{A} = \text{supp}(f) \cup \{\mathbf{0}\}$. We rewrite the SOS optimization problem (4.3) as

$$\begin{cases} \text{maximize} & \xi \\ \text{subject to} & f(\mathbf{x}) - \xi \in \Sigma(\mathscr{A}). \end{cases} \tag{4.6}$$

To exploit the sparsity, we replace the constraint $f(\mathbf{x}) - \xi \in \Sigma(\mathscr{A})$ by the stronger constraint $f(\mathbf{x}) - \xi \in \tilde{\Sigma}(\mathscr{A})$ to obtain

$$\begin{cases} \text{maximize} & \xi \\ \text{subject to} & f(\mathbf{x}) - \xi \in \tilde{\Sigma}(\mathscr{A}). \end{cases} \tag{4.7}$$

Let $\xi^*_{ssos}$ denote the optimal value of (4.7). Assume that $\mathscr{B} = \{\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_r\}$ and the cross sparsity pattern graph is $\tilde{G}(V_{\mathscr{A}}, \tilde{E}_{\mathscr{A}})$. Let $C_1, C_2, \ldots, C_t \subseteq V_{\mathscr{A}}$ denote the maximal cliques of $\tilde{G}(V_{\mathscr{A}}, \tilde{E}_{\mathscr{A}})$ and $\mathscr{B}_k = \{\boldsymbol{\omega}_i \in \mathscr{B} \mid i \in C_k\}, i = 1, 2, \ldots, t$. Then Theorem 3.1 allows us to decompose the single large SOS constraint $f(\mathbf{x}) - \xi \in \tilde{\Sigma}(\mathscr{A})$ into a set of SOS constraints with smaller dimensions,

$$\begin{cases} \text{maximize} & \xi \\ \text{subject to} & f(\mathbf{x}) - \xi \in \sum_{k=1}^t \mathbb{R}[\mathscr{B}_k]^2. \end{cases} \tag{4.8}$$

This can reduce the computational cost significantly if the size of the cliques $C_k, k = 1, \ldots, t$ is small.

The relation between the optimums of the polynomial optimization problem (4.2), the SOS optimization problem (4.3) and the sparse SOS optimization problem (4.7) are

$$\xi^* \geq \xi^*_{sos} \geq \xi^*_{ssos}.$$

## 5. When do $\Sigma(\mathscr{A})$ and $\tilde{\Sigma}(\mathscr{A})$ coincide

Given $\mathscr{A} \subseteq \mathbb{N}^n$ with $V(\mathscr{A}) \subseteq (2\mathbb{N})^n$, we define in Section 3.2 two sets of nonnegative polynomials: $\Sigma(\mathscr{A})$ and $\tilde{\Sigma}(\mathscr{A})$. Generally we have $\Sigma(\mathscr{A}) \supseteq \tilde{\Sigma}(\mathscr{A})$. There are certain cases in which $\Sigma(\mathscr{A}) = \tilde{\Sigma}(\mathscr{A})$ holds.

1.The quadratic case

Suppose $f \in \mathbb{R}[\mathbf{x}]$ is a quadratic polynomial. Let $M = [1, x_1, \ldots, x_n]$ be a monomial basis and assume $f = M^T Q M$ for a positive semidefinite matrix $Q = (q_{ij})$. Let $\mathbf{R} = (R_{ij})$ be the corresponding cross sparsity pattern matrix for $f$. If $x_i$ not appears in $f$, then we must have $q_{i0} = q_{0i} = 0$ and $R_{i0} = R_{0i} = 0$. If $x_i x_j$ not appears in $f$, then we must have $q_{ij} = q_{ji} = 0$ and $R_{ij} = R_{ji} = 0$. Therefore, $Q \in S_+^{n+1}(\tilde{E}_{\mathscr{A}}, 0)$, where $\mathscr{A} = \mathrm{supp}(f)$. So in this case we have $\Sigma(\mathscr{A}) = \tilde{\Sigma}(\mathscr{A})$.

2.$\mathscr{A} \subseteq (2\mathbb{N})^n$

Assume $\mathscr{B} = \{\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_r\}$ is as (3.3). In this case, the elements of the cross sparsity pattern matrix $\mathbf{R}_{\mathscr{A}}$ satisfy

$$R_{ij} = \begin{cases} 1, & \boldsymbol{\omega}_i + \boldsymbol{\omega}_j \in (2\mathbb{N})^n, \\ 0, & \text{otherwise.} \end{cases} \tag{5.1}$$

The corresponding cross sparsity pattern graph $G(V_{\mathscr{A}}, E_{\mathscr{A}})$ has $t$ connected components $C_1, C_2, \ldots, C_t$, everyone of which is a complete subgraph. Moreover, $i, j$ belong to the same connected component if and only if $\boldsymbol{\omega}_i + \boldsymbol{\omega}_j \in (2\mathbb{N})^n$.

Suppose $f(\mathbf{x}) \in \Sigma(\mathscr{A})$. We have $f(x_1, \ldots, x_i, \ldots, x_n) = f(x_1, \ldots, -x_i, \ldots, x_n)$ for $i = 1, \ldots, n$. It follows that the polynomial $f(\mathbf{x})$ has $n$ sign-symmetries defined by the $n$ standard basis vectors $\mathbf{e}_1, \ldots, \mathbf{e}_n$. Therefore, by Theorem 3 of [18], the monomials $\mathscr{B}$ can be block partitioned into $t$ blocks $\mathscr{B}_1, \ldots, \mathscr{B}_t$ where $\boldsymbol{\omega}_i, \boldsymbol{\omega}_j$ belong to the same block if and only if $\boldsymbol{\omega}_i + \boldsymbol{\omega}_j \in (2\mathbb{N})^n$, such that $f(\mathbf{x}) \in \sum_{k=1}^t \mathbb{R}[\mathscr{B}_k]^2$. Then $\mathscr{B}_k = \{\boldsymbol{\omega}_i \in \mathscr{B} \mid i \in C_k\}, i = 1, 2, \ldots, t$. So $f(\mathbf{x}) \in \tilde{\Sigma}(\mathscr{A})$ and hence $\Sigma(\mathscr{A}) = \tilde{\Sigma}(\mathscr{A})$.

**Remark 5.1.** *In* [18]*, sign-symmetries is exploited to block diagonalize sums of squares programming (Theorem 3). It is easy to show that our blocking decompositions are always a refinement of the block-diagonalizations obtained by sign-symmetries.*

## 6. Algorithms and Numerical results

In this section, we give examples and numerical results to illustrate the effectiveness of our method. It turns out that our method is extremely powerful and can deal with really huge polynomials that cannot be handled by other methods.

6.1. **Algorithms.** Determining cross sparse SOS polynomial can be easily divided into 4 steps:

(1) Computing support of SOS;
(2) Generate cross sparsity pattern graph;
(3) Computing the triangulation of graphs and get Blocked;
(4) Using SDP solver.

We simply use the computing connectivity branch to computing the triangulation. Here are some of the results of the polynomial. Note that because there are several ways to computing support of SOS, we do not calculate the time to computing support of SOS.

(CPU:Intel i7-4760HQ@2.10GHz(core 4,thread 8), memory:16G, SYSTEM:ARCH LINUX, SDP Solved:CSDP 6.2.0)

6.2. **SOS programming.**

6.2.1. Let $B_m = (\sum_{i=1}^{3m+2} x_i^2)((\sum_{i=1}^{3m+2} x_i^2)^2 - 2\sum_{i=1}^{3m+2} x_i^2 \sum_{j=1}^{m} x_{i+3j+1}^2)$, where $x_{3m+2+r} = x_r$. Note that $B_m$ is modified from [23]. For any $m \in \mathbb{N}^*$, $B_m$ is homogeneous and is a SOS polynomial. For these $B_m$'s, our algorithm SparseSOS dramatically reduces the problem sizes and the solving times (Table 1).

| m | #Basis | #Block | SOS | SparseSOS |
|---|---|---|---|---|
| 1 | 35 | $5\times 5, 10\times 1$ | 0.03s | 0.01s |
| 2 | 120 | $8\times 8, 56\times 1$ | 0.88s | 0.04s |
| 3 | 286 | $11\times 11, 165\times 1$ | 38.90s | 0.08s |
| 4 | 560 | $14\times 14, 364\times 1$ | 1001.40s | 0.24s |
| 5 | 969 | $17\times 17, 680\times 1$ | OOM | 0.65s |
| 6 | 1540 | $20\times 20, 1140\times 1$ | OOM | 1.63s |

TABLE 1. Sizes of PSD constraints and solving times before and after using SparseSOS for $B_m$'s. The notion $i \times j$ represents $i$ blocks of size $j$. The notion OOM indicates an out-of-memory error.

6.2.2. Monotone Column Permanent (MCP) Conjecture was given in [19]. In the dimension 4, this conjecture is equivalent to decide whether particular polynomials named by $p_{1,2}, p_{1,3}, p_{2,2}, p_{2,3}$ are nonnegative (the definitions of $p_{i,j}$ can be found in [20]). Actually, it was proved that every $p_{i,j}$ multiplied by a small particular polynomial is a SOS polynomial ([20]). Let

$$\begin{aligned} P_{1,2} &= (a^2 + 2b^2 + c^2) \cdot p_{1,2}, \\ P_{1,3} &= p_{1,3}, \\ P_{2,2} &= (a^2 + 2b^2 + c^2) \cdot p_{2,2}, \\ P_{2,3} &= (a^2 + 2b^2 + c^2) \cdot p_{2,3}. \end{aligned}$$

We use our algorithm SparseSOS to certify the nonnegativity of $P_{1,2}, P_{1,3}, P_{2,2}, P_{2,3}$. The result is listed in Table 2.

| | #Support | #Basis | #Block | SOS | SparseSOS |
|---|---|---|---|---|---|
| $P_{1,2}$ | 159 | 77 | $15, 2\times 12, 7\times 4, 3, 2\times 2, 3\times 1$ | 0.29s | 0.05s |
| $P_{1,3}$ | 53 | 29 | $8, 4\times 3, 2\times 2, 5\times 1$ | 0.29s | 0.02s |
| $P_{2,2}$ | 144 | 62 | $3\times 12, 2\times 4, 8\times 2, 2\times 1$ | 0.24s | 0.07s |
| $P_{2,3}$ | 107 | 53 | $2\times 10, 8, 4, 3, 8\times 2, 2\times 1$ | 0.12s | 0.05s |

TABLE 2. Sizes of PSD constraints and solving times before and after using SparseSOS for $P_{1,2}, P_{1,3}, P_{2,2}, P_{2,3}$.

6.2.3. The following polynomial Vor1 appears in [9]. It was proved that Vor1 is nonnegative and its discriminant (denoted by Vor2) is also nonnegative. The polynomial Vor2 has 1571 monomials and is of degree 30.

$$\begin{aligned} \text{Vor1} =& 16a^2(\alpha^2 + 1 + \beta^2)u^4 + 16a(-\alpha\beta a^2 + ax\alpha + 2a\alpha^2 + 2a + 2a\beta^2 + ay\beta - \\ & \alpha\beta)u^3 + ((24a^2 + 4a^4)\alpha^2 + (-24\beta a^3 - 24a\beta - 8ya^3 + 24xa^2 - 8ay) \end{aligned}$$

$$\alpha + 24a^2\beta^2 + 4\beta^2 - 8\beta x a^3 + 4y^2a^2 + 24y\beta a^2 - 8ax\beta + 16a^2 + 4x^2a^2)$$
$$u^2 + (-4\alpha a^3 + 4ya^2 - 4ax - 8a\alpha + 8\beta a^2 + 4\beta)(\beta - a\alpha + y - ax)u +$$
$$(a^2 + 1)(\beta - a\alpha + y - ax)^2$$

| | #Support | #Basis | #Block | SOS | SparseSOS |
|---|---|---|---|---|---|
| Vor1 | 63 | 18 | $2 \times 8, 1 \times 2$ | 0.01s | 0.01s |
| Vor2 | 1571 | 597 | $121, 88, 2 \times 70, 2 \times 64, 2 \times 60$ | 390.13s | 2.18s |

TABLE 3. Sizes of PSD constraints and solving times before and after using SparseSOS for Vor1, Vor2.

6.2.4. The polynomials $J_{40}, J_{421}, J_{50}, J_{521}$ (see the Appendix) are given by Jeffrey Uhlmann. The sizes of $J_{40}, J_{421}, J_{50}, J_{521}$ are listed in Table 3. One can see that $J_{40}, J_{50}$ are really huge and the corresponding SDPs are unsolvable. However, our algorithm SparseSOS can handle with them in less than one minute (Table 4).

| | #Support | dim | deg |
|---|---|---|---|
| $J_{40}$ | 138 | 6 | 12 |
| $J_{421}$ | 116 | 6 | 12 |
| $J_{50}$ | 5687 | 10 | 20 |
| $J_{521}$ | 5157 | 10 | 20 |

TABLE 4. Scales of $J_{40}$,$J_{421}$,$J_{50}$,$J_{521}$. The first column is the number of supports; the second column is the number of variables; the third column is the degrees.

| | #Basis | #Block | SOS | SparseSOS |
|---|---|---|---|---|
| $J_{40}$ | 64 | $14, 3 \times 10, 2 \times 6, 5, 3$ | 0.24s | 0.04s |
| $J_{421}$ | 48 | $12, 10, 7, 4, 3, 2, 3 \times 1$ | 0.13s | 0.04s |
| $J_{50}$ | 1014 | $121, 94, 92, 86, 84, 64, 62, 55,$ $2 \times 52, 51, 49, 45, 40, 39, 28$ | OOM | 44.65s |
| $J_{521}$ | 864 | $111, 80, 77, 76, 75, 55, 54, 52,$ $43, 2 \times 42, 39, 2 \times 34, 30, 20$ | OOM | 35.23s |

TABLE 5. Sizes of PSD constraints and solving times before and after using SparseSOS for $J_{40}$,$J_{421}$,$J_{50}$,$J_{521}$.

6.3. **Unconstrained polynomial optimization.**

6.4. **Comparison with minimal coordinate projections.** Permenter and Parrilo found block-diagonalizations for SOS programming using minimal coordinate projections in [25]. We compare the block-diagonalizations by minimal coordinate projections and the blocking decompositions by our method. In most cases, our blocking decompositions are a refinement of the block-diagonalizations by minimal coordinate projections.

Compare the size of blocks through the above examples.

## 7. Conclusions

## References


1. J. Agler, W. Helton, S. McCullough, L. Rodman, *Positive semidefinite matrices with a given sparsity pattern*, Linear algebra and its applications, 107(1988):101-149.
2. P. R. Amestoy, T. A. Davis, I. S. Duff, *Algorithm 837: AMD, an approximate minimum degree ordering algorithm*, ACM Transactions on Mathematical Software, 30(3)(2004):381-388.
3. A. Berry, J. R. S. Blair, P. Heggernes, B. W. Peyton, *Maximum cardinality search for computing minimal triangulations of graphs*, Algorithmica, 39(4)(2004):287-298.
4. D. Bertsimas, R. M. Freund, X. A. Sun, *An accelerated first-order method for solving SOS relaxations of unconstrained polynomial optimization problems*, Optim. Methods Softw., 28(3)(2013):424-441.
5. J. R. S. Blair, B. Peyton, *An introduction to chordal graphs and clique trees*, in Graph Theory and Sparse Matrix Computation, A. George, J. R. Gilbert, and J. W. H. Liu, eds., Springer-Verlag, New York, 1993:1¨C29.
6. J. S. Campos, P. Parpas, *A Multigrid Approach to SDP Relaxations of Sparse Polynomial Optimization Problems*, Siam Journal on Optimization, 28(1)2016:1-29.
7. L. Dai, B. Xia, *Smaller SDP for SOS decomposition*, Journal of Global Optimization, 63(2)(2015):343-361.
8. I. Z. Emiris, E. P. Tsigaridas, *Real algebraic numbers and polynomial systems of small degree*, Theoretical Computer Science, 409(2)(2008):186-199.
9. H. Everett, D. Lazard, S. Lazard, M. Safey El Din, *The voronoi diagram of three lines*, Discrete and Computational Geometry, 42(1)(2009):94-130.
10. D. R. Fulkerson, O. A. Gross, *Incidence matrices and interval graphs*, Pacific J. Math., 15(1965):835-855.
11. K. Gatermann, P. A. Parrilo, *Symmetry groups, semidefinite programs, and sums of squares*, Journal of Pure and Applied Algebra, 192(1)(2002):95-128.
12. M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.
13. P. Heggernes, *Minimal triangulations of graphs: a survey*, Discrete Mathematics, 306(3)(2006):297-317.
14. D. Henrion, J. Malick, *Projection methods in conic optimization*, in Handbook on Semidefinite, Conic and Polynomial Optimization, New York, NY, USA: Springer, 2012:565-600.
15. S. Iliman, T. de Wolff, *Amoebas, nonnegative poltnomials and sums of squares supported on circuits*, Res. Math. Sci., 3(2016), 3:9.
16. M. Kojima, S. Kim, H. Waki, *Sparsity in sums of squares of polynomials*, Math. Program., 103(2005):45-62.
17. J. Löfberg, YALMIP: a toolbox for modeling and optimization in MATLAB, In 2004 IEEE International Conference on Robotics and Automation (IEEE Cat. No.04CH37508), 284-289.
18. J. Löfberg, *Pre- and Post-Processing Sum-of-Squares Programs in Practice*, IEEE Transactions on Automatic Control, 54(5)(2009):1007-1011.
19. J. Haglund, K. Ono, D. G. Wagner, *Theorems and conjectures involving rook polynomials with real roots*, In: Proceedings of Topics in Number Theory and Combinatorics, 1997:207¨C221.
20. E. Kaltofen, B. Li, Z. Yang, L. Zhi, *Exact certification in global polynomial optimization via sums-of-squares of rational functions with rational coefficients*, Journal of Symbolic Computation, 47(1)(2012):1-15.

21. E. Kaltofen, Z. Yang, L. Zhi, *A proof of themonotone column permanent (mcp) conjecture for dimension 4 via sums-of-squares of rational functions*, In: Proceedings of the 2009 Conference on Symbolic Numeric Computation, SNC ¡¯09, 2009:65¨C70, ACM, New York.
22. B. Li, J. Nie, L. Zhi, *Approximate GCDs of polynomials and sparse SOS relaxations*, Theoretical Computer Science, 409(2)(2008):200-210.
23. A. Marandi, E. D. Klerk, J. Dahl, *Solving sparse polynomial optimization problems with chordal structure using the sparse bounded-degree sum-of-squares hierarchy*, Discrete Applied Mathematics, 2017.
24. P. A. Parrilo, *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*, Ph.D. Thesis, California Institute of Technology, 2000.
25. P. A. Parrilo, B. Sturmfels, *Minimizing Polynomial Functions*, Proceedings of the Dimacs Workshop on Algorithmic and Quantitative Aspects of Real Algebraic Geometry in Mathematics and Computer Science, 32(1)(2001):83-100.
26. F. Permenter, P. A. Parrilo, *Basis selection for SOS programs via facial reduction and polyhedral approximations*, Decision and Control. IEEE, 2014:6615-6620.
27. F. Permenter, P. A. Parrilo, *Finding sparse, equivalent SDPs using minimal coordinate projections*, In 54th IEEE Conference on Decision and Control, CDC 2015, Osaka, Japan, December 15-18, 2015:7274¨C7279.
28. B. Reznick, *Extremal PSD forms with few terms*, Duke Math. J., 45(1978):363-374.
29. L. Vandenberghe, M. S. Andersen, *Chordal Graphs and Semidefinite Optimization*, Now Publisher, 1900.
30. H. Waki, S. Kim, M. Kojima, M. Muramatsu, *Sums of squares and semidefinite program relaxations for polynomial optimization problems with structured sparsity*, SIAM Journal on Optimization, 17(1)(2016):218-242.
31. H. Waki, M. Muramatsu, *A facial reduction algorithm for finding sparse SOS representations*, Operations Research Letters, 38(5)(2009):361-365.
32. H. Waki, M. Muramatsu, *An extension of the elimination method for a sparse SOS polynomial*, Journal of the Operations Research Society of Japan, 4(4)(2017):161-190.
33. J. Wang, *Nonnegative Polynomials and Circuit Polynomials*, 2018, arXiv:1804.09455.
34. T. Weisser, J. B. Lasserre, K. C. Toh, *Sparse-BSOS: a bounded degree SOS hierarchy for large scale polynomial optimization with sparsity*, Mathematical Programming Computation, 10(1)(2018):1-32.
35. Z. Yang, G. Fantuzzi, A. Papachristodoulou, *Decomposition and completion of sum-of-squares matrices*, 2018, arXiv:1804.02711.
36. Z. Yang, G. Fantuzzi, A. Papachristodoulou, *Exploiting Sparsity in the Coefficient Matching Conditions in Sum-of-Squares Programming Using ADMM*, IEEE Control Systems Letters, 1(1)(2017):80-85.


# Appendices

$$
\begin{aligned}
J_{521} =&(ag+h)(-h+ci+ej-cfj)((bd+ce+f)(-f+ij-fj^2)(-(bd+ce+f)g\\
&(-g+ah+bi-aci+dj-aej-bfj+acfj)(-f+ij-fj^2)+d(bg+ch+\\
&i)(-i+fj)(-d+ae+bf-acf+gj-ahj-bij+acij-dj^2+aej^2+bfj^2\\
&-acfj^2))-(1+d^2+e^2+f^2)(1+j^2)(-(1+b^2+c^2)g(-g+ah+bi-aci\\
&+dj-aej-bfj+acfj)(1+f^2+i^2-2fij+f^2j^2)+b(bg+ch+i)(-i+\\
&fj)(-b+ac+df-aef-bf^2+acf^2+gi-ahi-bi^2+aci^2-fgj+afhj\\
&-dij+aeij+2bfij-2acfij+dfj^2-aefj^2-bf^2j^2+acf^2j^2))-j(dg+\\
&eh+fi+j)(-(1+b^2+c^2)d(-d+ae+bf-acf+gj-ahj-bij+acij-\\
&dj^2+aej^2+bfj^2-acfj^2)(1+f^2+i^2-2fij+f^2j^2)+b(bd+ce+f)(-f
\end{aligned}
$$

$$+ij-fj^2)(-b+ac+df-aef-bf^2+acf^2+gi-ahi-bi^2+aci^2-fgj+afhj-dij+aeij+2bfij-2acfij+dfj^2-aefj^2-bf^2j^2+acf^2j^2)))-(bg+ch+i)(-i+fj)((ad+e)(-e+cf+hj-cij-ej^2+cfj^2)(-(bd+ce+f)g(-g+ah+bi-aci+dj-aej-bfj+acfj)(-f+ij-fj^2)+d(bg+ch+i)(-i+fj)(-d+ae+bf-acf+gj-ahj-bij+acij-dj^2+aej^2+bfj^2-acfj^2))-(1+d^2+e^2+f^2)(1+j^2)(-(ab+c)g(-g+ah+bi-aci+dj-aej-bfj+acfj)(-c+ef-cf^2+hi-ci^2-fhj-eij+2cfij+efj^2-cf^2j^2)+a(bg+ch+i)(-i+fj)(-a+bc-ac^2+de-ae^2-cdf-bef+2acef+bcf^2-ac^2f^2+gh-ah^2-cgi-bhi+2achi+bci^2-ac^2i^2-egj+cfgj-dhj+2aehj+bfhj-2acfhj+cdij+beij-2aceij-2bcfij+2ac^2fij+dej^2-ae^2j^2-cdfj^2-befj^2+2acefj^2+bcf^2j^2-ac^2f^2j^2))-j(dg+eh+fi+j)(-(ab+c)d(-d+ae+bf-acf+gj-ahj-bij+acij-dj^2+aej^2+bfj^2-acfj^2)(-c+ef-cf^2+hi-ci^2-fhj-eij+2cfij+efj^2-cf^2j^2)+a(bd+ce+f)(-f+ij-fj^2)(-a+bc-ac^2+de-ae^2-cdf-bef+2acef+bcf^2-ac^2f^2+gh-ah^2-cgi-bhi+2achi+bci^2-ac^2i^2-egj+cfgj-dhj+2aehj+bfhj-2acfhj+cdij+beij-2aceij-2bcfij+2ac^2fij+dej^2-ae^2j^2-cdfj^2-befj^2+2acefj^2+bcf^2j^2-ac^2f^2j^2)))-j(dg+eh+fi+j)((ad+e)(-e+cf+hj-cij-ej^2+cfj^2)(-(1+b^2+c^2)g(-g+ah+bi-aci+dj-aej-bfj+acfj)(1+f^2+i^2-2fij+f^2j^2)+b(bg+ch+i)(-i+fj)(-b+ac+df-aef-bf^2+acf^2+gi-ahi-bi^2+aci^2-fgj+afhj-dij+aeij+2bfij-2acfij+dfj^2-aefj^2-bf^2j^2+acf^2j^2))-(bd+ce+f)(-f+ij-fj^2)(-(ab+c)g(-g+ah+bi-aci+dj-aej-bfj+acfj)(-c+ef-cf^2+hi-ci^2-fhj-eij+2cfij+efj^2-cf^2j^2)+a(bg+ch+i)(-i+fj)(-a+bc-ac^2+de-ae^2-cdf-bef+2acef+bcf^2-ac^2f^2+gh-ah^2-cgi-bhi+2achi+bci^2-ac^2i^2-egj+cfgj-dhj+2aehj+bfhj-2acfhj+cdij+beij-2aceij-2bcfij+2ac^2fij+dej^2-ae^2j^2-cdfj^2-befj^2+2acefj^2+bcf^2j^2-ac^2f^2j^2))-j(dg+eh+fi+j)(-b(ab+c)(-c+ef-cf^2+hi-ci^2-fhj-eij+2cfij+efj^2-cf^2j^2)(-b+ac+df-aef-bf^2+acf^2+gi-ahi-bi^2+aci^2-fgj+afhj-dij+aeij+2bfij-2acfij+dfj^2-aefj^2-bf^2j^2+acf^2j^2)+a(1+b^2+c^2)(1+f^2+i^2-2fij+f^2j^2)(-a+bc-ac^2+de-ae^2-cdf-bef+2acef+bcf^2-ac^2f^2+gh-ah^2-cgi-bhi+2achi+bci^2-ac^2i^2-egj+cfgj-dhj+2aehj+bfhj-2acfhj+cdij+beij-2aceij-2bcfij+2ac^2fij+dej^2-ae^2j^2-cdfj^2-befj^2+2acefj^2+bcf^2j^2-ac^2f^2j^2))-(1+g^2+h^2+i^2+j^2)((ad+e)(-e+cf+hj-cij-ej^2+cfj^2)(-(1+b^2+c^2)d(-d+ae+bf-$$

$$acf+gj-ahj-bij+acij-dj^2+aej^2+bfj^2-acfj^2)(1+f^2+i^2-2fij+f^2j^2)+b(bd+ce+f)(-f+ij-fj^2)(-b+ac+df-aef-bf^2+acf^2+gi-ahi-bi^2+aci^2-fgj+afhj-dij+aeij+2bfij-2acfij+dfj^2-aefj^2-bf^2j^2+acf^2j^2))-(bd+ce+f)(-f+ij-fj^2)(-a+bc-ac^2+de-ae^2-cdf-bef+2acef+bcf^2-ac^2f^2+gh-ah^2-cgi-bhi+2achi+bci^2-ac^2i^2-egj+cfgj-dhj+2aehj+bfhj-2acfhj+cdij+beij-2aceij-2bcfij+2ac^2fij+dej^2-ae^2j^2-cdfj^2-befj^2+2acefj^2+bcf^2j^2-ac^2f^2j^2))+(1+d^2+e^2+f^2)(1+j^2)(-b(ab+c)(-c+ef-cf^2+hi-ci^2-fhj-eij+2cfij+efj^2-cf^2j^2)(-b+ac+df-aef-bf^2+acf^2+gi-ahi-bi^2+aci^2-fgj+afhj-dij+aeij+2bfij-2acfij+dfj^2-aefj^2-bf^2j^2+acf^2j^2)+a(1+b^2+c^2)(1+f^2+i^2-2fij+f^2j^2)(-a+bc-ac^2+de-ae^2-cdf-bef+2acef+bcf^2-ac^2f^2+gh-ah^2-cgi-bhi+2achi+bci^2-ac^2i^2-egj+cfgj-dhj+2aehj+bfhj-2acfhj+cdij+beij-2aceij-2bcfij+2ac^2fij+dej^2-ae^2j^2-cdfj^2-befj^2+2acefj^2+bcf^2j^2-ac^2f^2j^2)))$$

Jie Wang, School of Mathematical Sciences, Peking University
*Email address*: wangjie212@pku.edu.cn