

5-2022

Modern Theory of Copositive Matrices

Yuqiao Li
William & Mary

Follow this and additional works at: <https://scholarworks.wm.edu/honorstheses>



Part of the [Algebra Commons](#), and the [Discrete Mathematics and Combinatorics Commons](#)

Recommended Citation

Li, Yuqiao, "Modern Theory of Copositive Matrices" (2022). *Undergraduate Honors Theses*. William & Mary. Paper 1866.

<https://scholarworks.wm.edu/honorstheses/1866>

This Honors Thesis -- Open Access is brought to you for free and open access by the Theses, Dissertations, & Master Projects at W&M ScholarWorks. It has been accepted for inclusion in Undergraduate Honors Theses by an authorized administrator of W&M ScholarWorks. For more information, please contact scholarworks@wm.edu.

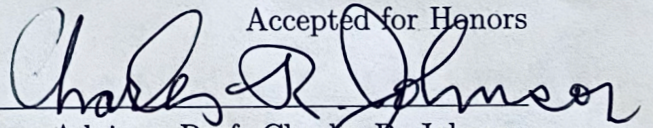
Modern Theory of Copositive Matrices

A thesis submitted in partial fulfillment of the requirement
for the degree of Bachelor of Science in
Mathematics from the College of William & Mary in Virginia,

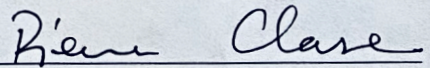
by

Yuqiao Li

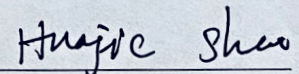
Accepted for Honors



Advisor: Prof. Charles R. Johnson



Prof. Pierre Clare



Prof. Huajie Shao

Williamsburg, Virginia

May 6 2022

Contents

Abstract	v
1 Introduction	1
1.1 Notations & Symbols	2
1.2 Definitions	2
1.3 Background	4
1.4 Outline & Organization	4
2 Basic Properties of CoP Matrices	5
2.1 Basic Properties	5
2.2 Exceptional vs. Ordinary	6
2.3 Spectral Properties	8
2.4 Schur Complements	8
3 Checking Copositivity	9
3.1 For matrices in $M_n(\{1, 0, -1\})$	9
3.2 For matrices with non-positive off-diagonal entries	9
3.3 General case: Kaplan's Theorem	10
4 Dual Cone: Completely Positive (CP) Matrices	11
4.1 Completely Positive Matrices	12

4.2	Hadamard Product & Kronecker Product	12
5	The Ordinary Recognition & Decomposition Problem	14
5.1	Failure of the naive decomposition	14
5.2	From Ordinary Decomposition to PSD Completion	15
5.3	Odd powers of exceptional copositive matrices	17
5.4	3-by-3 Ordinary Decomposition	18
5.4.1	Decomposing a 3-by-3 CoP matrix	18
5.4.2	Observations on eigenvalues and eigenvectors	20
6	Ordinary Copositive Matrices and their Graphs	22
6.1	Preliminaries	22
6.2	Properties Ordinary Graphs	23
6.3	Specific ordinary graphs	25
6.4	Ordinary graphs on 5 vertices	25
6.5	Ordinary graphs on 6 vertices	26
6.6	Python Codes for Checking Copositivity	29
6.6.1	5-by-5 CoP Checking	29
6.6.2	6-by-6 CoP Checking	31
6.6.3	7-by-7 CoP Checking	34

Abstract

Copositivity is a generalization of positive semidefiniteness. It has applications in theoretical economics, operations research, and statistics. An n -by- n real, symmetric matrix A is copositive (CoP) if $x^T Ax \geq 0$ for any nonnegative vector $x \geq 0$. The set of all CoP matrices forms a convex cone. A CoP matrix is ordinary if it can be written as the sum of a positive semidefinite (PSD) matrix and a symmetric nonnegative (sN) matrix. When $n < 5$, all CoP matrices are ordinary. However, recognizing whether a given CoP matrix is ordinary and determining an ordinary decomposition (PSD + sN) is still an unsolved problem. Here, we give an overview on modern theory of CoP matrices, talk about our progress on the ordinary recognition and decomposition problem, and emphasize the graph theory aspect of ordinary CoP matrices.

Chapter 1

Introduction

Copositive (CoP) matrices are generalizations of positive semidefinite (PSD) matrices. They have applications in optimization, game theory, etc. In this paper, we give a survey on the class of copositive matrices, discuss properties related to its dual structure and its graphs, and present the ordinary recognition and decomposition problem.

Our motivation for studying this subject comes from the fact that copositivity is a generalization of positive semidefiniteness. Specifically, we wonder what properties of PSD matrices can be carried over to CoP matrices. For example, positive semidefiniteness is preserved under taking product, Hadamard product, and Kronecker products; we examine Hadamard products and Kronecker products on CoP matrices in Ch. 4. PSD matrices are preserved under polynomial p such that $p(t) \geq 0 \forall t \geq 0$, so we also wonder what polynomial preserves copositivity, and we discuss this in Ch. 5.

In general, since the set of all PSD matrices are a small subset of the set of all CoP matrices, many PSD properties do not apply to CoP matrices. Therefore, we include a few counterexamples and remarks to illustrate the difference between the class of PSD and CoP matrices.

1.1 Notations & Symbols

In this paper, we will be using the abbreviations of terminologies shown in Table 1 and notations and symbols shown in Table 2. These are the standard matrix notation presented in [1, 2, 3].

Table 1.1: Abbreviations

CoP	Copositive
SCoP	Strictly copositive
CP	Completely positive
PD	Positive definite
PSD	Positive semidefinite
sN	Symmetric nonnegative

1.2 Definitions

To begin our discussion, we first state the following definitions.

Definition 1.2.1. Let $A \in M_n(\mathbb{R})$ be a symmetric matrix, and let $x \in \mathbb{R}^n$ be a vector. Then, the *quadratic form* of A , $Q(A)$, is defined as $Q(x) = x^T Ax$.

Definition 1.2.2. Let $A \in M_n(\mathbb{R})$ be a symmetric matrix. Then, A is *positive definite* (PD) if the quadratic form $x^T Ax$ is positive for all nonzero vectors $x \in \mathbb{R}^n$. Similarly, A is *positive semidefinite* (PSD) if the quadratic form $x^T Ax$ is nonnegative for all nonzero vectors $x \in \mathbb{R}^n$.

As a generalization of PSD matrices, we define copositive matrices as below.

Definition 1.2.3. Let $A \in M_n(\mathbb{R})$ be a symmetric matrix, and let \mathbb{R}_+^n be the set of all vectors $x \in \mathbb{R}^n$ with nonnegative entries. Then, A is *copositive* (CoP) if its quadratic form $Q(x) = x^T Ax \geq 0$ for all nonzero vectors $x \in \mathbb{R}_+^n$. Similarly, A is *strictly copositive* if $Q(x) = x^T Ax > 0$ for all nonzero vectors $x \in \mathbb{R}_+^n$.

Table 1.2: Notations and Symbols

\mathbb{R}	Fields of real numbers
\mathbb{R}^n	All column n -vectors of real numbers
\mathbb{R}_+	All real, nonnegative numbers
\mathbb{R}_+^n	All vectors in \mathbb{R}^n with nonnegative entries
$M_{m,n}(\mathbb{F})$	All m -by- n matrices over field \mathbb{F}
$M_n(\mathbb{F})$	All n -by- n matrices over field \mathbb{F}
$M_n(\{1, 0, -1\})$	All n -by- n matrices with entries in $\{1, 0, -1\}$
$M_n(\{1, -1\})$	All n -by- n matrices with entries in $\{1, -1\}$
A^T, A^*	Transpose and conjugate transpose of a matrix A
$A[\alpha]$	Principal submatrix whose rows and columns are indexed by set α
a_{ij}	The (i, j) entry of a matrix $A = [a_{ij}]$
e_i	The column vector with the i^{th} entry being 1 and others being zero.
$Q(x) = x^T A x$	Quadratic form of matrix A
$A \circ B$	Hadamard product of matrices A and B
$A \otimes B$	Kronecker product of matrices A and B
$\sigma(A)$	Spectrum (eigenvalues) of $A \in M_n(\mathbb{F})$
$\text{Tr}(A)$	Trace of a matrix $A \in M_n(\mathbb{F})$
$\rho(A)$	Spectral radius of A
K^*	Dual cone of a cone K
\mathcal{C}_n	The set of all CoP matrices in $M_n(\mathbb{R})$.
\mathcal{P}_n	The set of all PSD matrices in $M_n(\mathbb{R})$.
\mathcal{N}_n	The set of all sN matrices in $M_n(\mathbb{R})$.
\mathcal{C}_n^*	The set of all CP matrices in $M_n(\mathbb{R})$.

Definition 1.2.4. Let C be a subset of a vector space. For positive scalars a, b , if $ax + by \in C$ for any $x, y \in C$, then C is a *convex cone*.

We note that the set of all CoP matrices, denoted by \mathcal{C}_n , forms a convex cone. Let $A, B \in \mathcal{C}_n$ with quadratic forms $Q_A(x), Q_B(x)$, respectively. Then, for all nonzero vectors $x \in \mathbb{R}_+^n$ and any $\alpha, \beta \in \mathbb{R}_+$, $Q_{\alpha A + \beta B}(x) = \alpha \cdot Q_A(x) + \beta \cdot Q_B(x) \geq 0$. Thus, $\alpha A + \beta B \in \mathcal{C}_n$, and \mathcal{C}_n is closed under taking positive linear combinations. For further discussion of the cone structure of the set of all CoP matrices, see [5].

1.3 Background

The idea of copositivity was first defined by Motzkin in 1952 as generalizations of positive semidefiniteness [4]. CoP matrices have a wide range of potential applications. In optimization theory, copositivity offers a unified convex way to reformulate non-convex mixed quadratic programs into convex programs [5]. Besides optimizations, CoP matrices also have application in differential equations [6] and theoretical economics [7]. Generally speaking, CoP matrices offer strong modeling power. However, Kaplan shows that the process of checking copositivity for any given matrix requires some efforts, as it is NP-hard (with more detail described in Sec. 3.3). Therefore, more studies are needed in this novel area.

1.4 Outline & Organization

In this paper, we give a survey on modern theory of CoP matrices and discuss our progress on the ordinary recognition and decomposition problem. In Ch. 2, we will discuss basic properties of CoP matrices, such as some entry-wise properties, exceptional and ordinary CoP matrices, spectral properties, and the Schur complement of CoP matrices. In Ch. 3, we will mention three methods of checking copositivity for different classes of matrices. In Ch. 4, we will discuss the dual cone of CoP matrices. In Ch. 5, we propose the ordinary recognition and decomposition problem. In Ch. ??, we will examine ordinary CoP matrices from the graph theory aspect.

Chapter 2

Basic Properties of CoP Matrices

Lemma 2.0.1. *Let $A \in M_n(\mathbb{R})$ be a CoP matrix. Then, all principal submatrices of A is CoP.*

Proof. Proof by contradiction: suppose there exists a principal submatrix $A[\alpha]$ of a CoP matrix $A \in M_n(\mathbb{R})$ that is not CoP, with $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_k\} \subseteq \{1, 2, \dots, n\}$ as the index set of the row and column of this principal submatrix. Then, there exists a vector $x \in \mathbb{R}_+^k, x \neq 0$ such that $Q(x) = x^T A[\alpha]x < 0$. We extend this vector $x \in \mathbb{R}_+^k$ to a vector $x' \in \mathbb{R}_+^n$ by keeping all entries in x at the indexed place according to α , and the other entries of x' are zeros. Then, we have $x'^T A x' = x^T A[\alpha]x < 0$ for this nonnegative vector $x' \neq 0$. Thus, A is no longer CoP, contradicting the given condition. □

2.1 Basic Properties

In[8], Bundfuss presents the following four entrywise properties for CoP matrices.

Proposition 2.1.1. *Let $A = [a_{ij}]$ be a CoP matrix. Then:*

- (i) $a_{ii} \geq 0$ for all i .
- (ii) If $a_{ii} = 0$, then $a_{ij} \geq 0$ for all j .

(iii) $a_{ij} \geq -\sqrt{a_{ii}a_{jj}}$ for all i and all j .

Proof. (i). We may think of a_{ii} as a 1-by-1 principal submatrix of A . Since the quadratic form $Q(x)$ for $x \in \mathbb{R}_+^1$ is nonnegative, a_{ii} must be nonnegative.

(ii). Proof by contradiction: suppose there exists $i, j \in \{1, \dots, n\}$ such that $a_{ii} = 0, a_{ij} < 0$. From (i), we know $a_{jj} = 0$. Let $e_i \in \mathbb{R}^n$ be the column vector with i^{th} entry being 1 and others being zero. Then, the vector $x = (a_{jj} + 1)e_i - a_{ij}e_j$ is in \mathbb{R}_+^n . Consider the quadratic form of x : $Q(x) = ((a_{jj} + 1)e_i - a_{ij}e_j)^T A((a_{jj} + 1)e_i - a_{ij}e_j) = -(a_{jj} + 2)a_{ij}^2 < 0$. However, A is CoP and $x \in \mathbb{R}_+^n$. Thus, we have a contradiction, and $a_{ij} \geq 0$.

(iii). Proof by contradiction: suppose there exists $i, j \in \{1, \dots, n\}$ such that $a_{ij} < -\sqrt{a_{ii}a_{jj}}$. Thus, from this condition, $a_{ij} < 0$ and $a_{ij}^2 > a_{ii} \cdot a_{jj}$. Since $a_{ij} < 0$, from (i) and (ii), $a_{ii} > 0$. Then, the vector $x = a_{jj}e_i - a_{ij}e_i$ is in \mathbb{R}_+^n . Consider the quadratic form of x : $Q(x) = (a_{jj}e_i - a_{ij}e_i)^T A(a_{jj}e_i - a_{ij}e_i) = a_{jj}(a_{ii}a_{ij} - a_{ij}^2) < 0$. Since A is CoP and $x \in \mathbb{R}_+^n$, we have a contradiction, and $a_{ij} \geq -\sqrt{a_{ii}a_{jj}}$. \square

We note that this proposition offers us convenience when dealing with CoP matrices with unknown off-diagonal entries. Specifically, if the diagonal entry is zero, the off-diagonal entries can only be nonnegative. Furthermore, we can use diagonal congruence to scale its nonzero diagonal entries to 1, which may simplify some calculations.

2.2 Exceptional vs. Ordinary

First, we define the following sets of matrices. Let $\mathcal{C}_n = \{A \in M_n(\mathbb{R}) : A \text{ is copositive (CoP)}\}$; $\mathcal{P}_n = \{A \in M_n(\mathbb{R}) : A \text{ is positive semidefinite (PSD)}\}$; $\mathcal{N}_n = \{A \in M_n(\mathbb{R}) : A \text{ is symmetric and entry-wise nonnegative (sN)}\}$.

Definition 2.2.1. A CoP matrix is called *exceptional* if it is not the sum of a PSD

matrix and an sN matrix. Otherwise, we say the CoP matrix is *ordinary* (in some literature, ordinary CoP matrix is also called SPN).

In [9], Diananda proves the following theorem.

Theorem 2.2.2. *In general, $\mathcal{P}_n + \mathcal{N}_n \subset \mathcal{C}_n$. For $n \leq 4$, $\mathcal{P}_n + \mathcal{N}_n = \mathcal{C}_n$.*

Let $A \in M_n(\mathbb{R})$ be a CoP matrix. Then, we outline the specific type of A as the following:

- $n = 1$: $A \in \mathbb{R}_+ \cup \{0\}$;
- $n = 2$: $A \in \mathcal{P}_n$ or $A \in \mathcal{N}_n$;
- $n = 3$ or 4 : $A \in \mathcal{P}_n + \mathcal{N}_n$;
- $n \geq 5$: $A \in \mathcal{P}_n + \mathcal{N}_n$, or A is exceptional.

Here, we provide examples to help illustrate the case of ordinary and exceptional CoP matrix. We will discuss more in-depth of ordinary and exceptional CoP matrices in Ch. 5, and the graph of ordinary matrices in Ch. 6.

Example 2.2.3. In this example, A is ordinary CoP, and A can be decomposed into PSD + sN.

$$A = \begin{bmatrix} 1 & 3 & -1 \\ 3 & 1 & 1 \\ -1 & 3 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 1 \\ -1 & 1 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 3 & 0 \\ 3 & 0 & 2 \\ 0 & 2 & 1 \end{bmatrix}.$$

Example 2.2.4. The following matrix B is known as the *Horn matrix*. It is exceptional CoP and is not a sum of PSD and sN.

$$B = \begin{bmatrix} 1 & -1 & 1 & 1 & -1 \\ -1 & 1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & 1 & -1 \\ -1 & 1 & 1 & -1 & 1 \end{bmatrix}$$

2.3 Spectral Properties

We further note the following spectral property of CoP matrices.

Definition 2.3.1. The *spectral radius*, $\rho(A)$, of a matrix A is the largest absolute value of A 's eigenvalues.

In [10], Haynsworth et al. show that CoP matrices have the Perron property.

Theorem 2.3.2. *Let A be CoP. Then, the spectral radius, $\rho(A)$, is an eigenvalue of A .*

2.4 Schur Complements

In [11], Johnson et al. provide a survey on closure of matrix classes under Schur Complementation. Specifically, the class of CoP matrix is not closed under taking Schur complement and taking inverses; it is closed under inverse class Schur complementation.

Chapter 3

Checking Copositivity

3.1 For matrices in $M_n(\{1, 0, -1\})$

For CoP matrices with entries from $\{1, -1\}$ or $\{1, 0, -1\}$, there is an easier way to check copositivity [3].

Theorem 3.1.1. *Let $A \in M_n(\mathbb{R})$ be a symmetric matrix. Then, A is CoP if and only if all of its principal submatrices do not contain the following “forbidden” patterns (up to permutation similarity):*

$$\begin{bmatrix} 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \end{bmatrix} \quad \& \quad \begin{bmatrix} 1 & -1 & -1 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}.$$

A proof of this theorem can be found in [3].

3.2 For matrices with non-positive off-diagonal entries

Theorem 3.2.1. *Let $A \in M_n(\mathbb{R})$ be a symmetric matrix with non-positive off-diagonal entries. Then, the following four conditions are equivalent:*

1. A is a Z-matrix;
2. A is PSD;

3. A is an M -matrix;

4. A is CoP .

3.3 General case: Kaplan's Theorem

We now introduce the following method to check copositivity for any given symmetric matrix [12].

Theorem 3.3.1. (*Kaplan's Theorem*)

Let $A \in M_n(\mathbb{R})$ be a symmetric matrix. Then A is CoP if and only if every principal submatrix B of A has no eigenvector $v > 0$ with associated eigenvalue $\lambda < 0$.

Proof. Proof: to be completed. □

This theorem provides us with a systematic way to check copositivity, but it also shows that checking copositivity is NP-hard. In appendix, we present codes for a computer algorithm to check copositivity using Kaplan's theorem.

Chapter 4

Dual Cone: Completely Positive (CP) Matrices

Before we begin our discussion, we would like to outline our motivation of studying the dual cone of CoP matrices. Since CoP matrices are generalizations of PSD matrices, we wonder what matrix operations preserve copositivity. First, if we simply multiply two PSD matrices, the product is PSD. However, multiplying two CoP matrices does not give a CoP matrix, as the counterexample below.

Example 4.0.1. Let $A = \begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{bmatrix}$, $B = \begin{bmatrix} 1 & 1 & -1 \\ 1 & 1 & 1 \\ -1 & 1 & 1 \end{bmatrix}$. Here, A, B are both CoP according to Thm. 3.1.1. Then, the product of A and B is

$$A \cdot B = \begin{bmatrix} -1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 1 & -1 \end{bmatrix}.$$

The diagonals of AB are negative, so AB cannot be a CoP matrix.

In addition to multiplication, the Kronecker product and Hadamard product of two PSD matrices are PSD. Therefore, we wonder if Kronecker product and Hadamard product also preserves copositivity. The following counterexample shows that if we have two CoP matrices in general, then their Hadamard product may not be CoP. However, if we have one CoP matrix and one CP matrix, then their Hadamard and Kronecker products are CoP.

4.1 Completely Positive Matrices

To begin our discussion on the dual of CoP matrices, we state the following definitions.

Definition 4.1.1. Let $M_{n,m}$ be the class of all n -by- m matrices. Let $A \in \mathbb{S}_n$. A is *completely positive (CP)* if A can be factorized as $A = BB^T$ for nonnegative matrix $B \in M_{n,m}$. CP matrices are special PSD matrices, while CP matrices are also CoP.

Definition 4.1.2. For a cone $K \subset \mathbb{S}_n$, the dual cone is: $K^* = \{Y \in \mathbb{S}_n : \text{Tr}(Y^T X) \geq 0 \forall X \in K\}$.

The set of all CP matrices also form a convex cone. Furthermore, since $x^T A x = \text{Tr}(A^T x x^T)$, all matrices of the form $x x^T$ with $x \geq 0$ are in the dual cone of \mathcal{C}_n . Thus, the cone of all CP matrices is the dual of the cone of all CoP matrices.

4.2 Hadamard Product & Kronecker Product

Positive semidefiniteness is preserved under Hadamard and Kronecker products. Since CoP matrices are generalizations of PSD matrices, we wonder if copositivity would also be preserved. We first present the definitions of Hadamard and Kronecker products in the following. For more information on Kronecker and Hadamard products, see [3, 2, 1].

Definition 4.2.1. Let M_n be the class of all n -by- n matrices. Let $A = (a_{ij}), B = (b_{ij}), A, B \in M_n$. The *Hadamard product* of A and B , denoted by $A \circ B$, is an entrywise multiplication: $A \circ B = (a_{ij} b_{ij})$.

Definition 4.2.2. Let $A = (a_{ij}) \in M_{m,n}, B \in M_{p,q}$. Then, the *Kronecker product* of

A and B , denoted by $A \otimes B$, is a pm -by- qn matrix:

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & \cdots & a_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathbf{B} & \cdots & a_{mn}\mathbf{B} \end{bmatrix}$$

Example 4.2.3. This example shows that if we have two CoP matrices in general, their Hadamard product may not be CoP.

$$\begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{bmatrix} \circ \begin{bmatrix} 1 & 1 & -1 \\ 1 & 1 & 1 \\ -1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \end{bmatrix}$$

We use the checking CoP method stated in Sec. 3.1. The first two matrices are both CoP, but their Hadamard product is not CoP.

Lemma 4.2.4. *Let $A, B \in M_n(\mathbb{F})$. Then, $A \circ B$ is a principal submatrix of $A \otimes B$.*

From the definition, we remark that for two matrices $A, B \in M_n$, $A \circ B$ is a principal submatrix of $A \otimes B$. Therefore, if $A \circ B$ is not a CoP matrix, then $A \otimes B$ is not a CoP matrix.

In [13], Bomze et al. provide the following theorems that state the circumstances where Hadamard and Kronecker products preserve copositivity.

Theorem 4.2.5. *Let $A \in \mathbb{S}_n$ be a CoP matrix. Then, $B \circ A$ is copositive if and only if B is CP; $B \otimes A$ is copositive if and only if B is CP.*

Proof. A proof of this theorem can be found in [14]. □

Chapter 5

The Ordinary Recognition & Decomposition Problem

In this chapter, we will discuss the ordinary recognition & decomposition problem and our progress on this problem. First, we state the problem as the following:

Problem 5.0.1. (*The Ordinary Recognition & Decomposition Problem*)

- 1) Given a CoP matrix, how to recognize whether it is ordinary or not?
- 2) If we have an ordinary CoP matrix, how to decompose it into PSD + sN?

5.1 Failure of the naive decomposition

First, we want to remark the failure of the “naive” decomposition. By naive decomposition, we mean the most straightforward way one could think of to decompose a matrix into PSD + sN. In particular, all negative entries can only go to the PSD part. Since increasing the diagonal entries does not destroy the positive semi-definiteness, we may keep all diagonal entries in the PSD part. As in the naive decomposition, we want to take anything else left to be in the sN part.

In the following, we provide a counterexample that the naive decomposition fails to provide a PSD + sN decomposition.

Example 5.1.1. (*Failure of the naive decomposition*)

Let A be the following CoP matrix. Since A is 3-by-3, A is ordinary.

$$A = \begin{bmatrix} 13 & -9 & 28 \\ -9 & 12 & -11 \\ 28 & -11 & 20 \end{bmatrix}$$

If we decompose A according to the naive decomposition, then we will have:

$$A = \begin{bmatrix} 13 & -9 & 0 \\ -9 & 12 & -11 \\ 0 & -11 & 20 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 28 \\ 0 & 0 & 0 \\ 28 & 0 & 0 \end{bmatrix}.$$

We note that the first part, $B = \begin{bmatrix} 13 & -9 & 0 \\ -9 & 12 & -11 \\ 0 & -11 & 20 \end{bmatrix}$, is not PSD; it has an eigenvalue of -0.16 . Therefore, the naive decomposition does not work.

We would also like to remark the subtlety of the ordinary decomposition problem by presenting a possible ordinary decomposition for the matrix A . In particular, if we transfer a little amount of positive weight to the zero entries of B , we can obtain a PSD matrix: $B' = \begin{bmatrix} 13 & -9 & 1 \\ -9 & 12 & -11 \\ 1 & -11 & 20 \end{bmatrix}$. Therefore, a possible ordinary decomposition of A is:

$$A = \begin{bmatrix} 13 & -9 & 1 \\ -9 & 12 & -11 \\ 1 & -11 & 20 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 27 \\ 0 & 0 & 0 \\ 27 & 0 & 0 \end{bmatrix}.$$

5.2 From Ordinary Decomposition to PSD Completion

We claim that the ordinary recognition problem is equivalent to a restricted type of PSD completion problem. Let $A, N, P \in M_n(\mathbb{R})$ be symmetric, and let A be the ordinary CoP matrix that we want to decompose into PSD + sN. Let N contain only the negative off-diagonal entries of A , and let P contain only the positive off-diagonal entries of A .

Now, let $B = P + N$, and let the zero entries in B be unspecified. Then, we make the following conjecture.

Conjecture 5.2.1. Let A, B, P be the matrices constructed as above. Then, A is ordinary if and only if B has a PSD completion with entries no more than P .

With this conjecture, we translate the ordinary recognition problem to a PSD completion problem. Specifically, we start with a “skeleton” B , and we gradually transfer positive weight from P to B . If we use up everything but still not yet gotten a PSD matrix, then the original matrix A is exceptional.

$$A = \begin{bmatrix} a & -d & f \\ -d & b & -e \\ f & -e & c \end{bmatrix}; \Rightarrow \text{”Skeleton” } B = \begin{bmatrix} a & -d & 0 \\ -d & b & -e \\ 0 & -e & c \end{bmatrix}$$

To illustrate the procedure of our ordinary recognition method, we include the example below. As stated in prior section, the Horn matrix is an exceptional CoP matrix.

Example 5.2.2. Horn matrix:

$$A = \begin{bmatrix} 1 & -1 & 1 & 1 & -1 \\ -1 & 1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & 1 & -1 \\ -1 & 1 & 1 & -1 & 1 \end{bmatrix};$$

The decomposed D, P , and skeleton B are:

$$D = I_5, P = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix}; B = \begin{bmatrix} 1 & -1 & 0 & 0 & -1 \\ -1 & 1 & -1 & 0 & 0 \\ 0 & -1 & 1 & -1 & 0 \\ 0 & 0 & -1 & 1 & -1 \\ -1 & 0 & 0 & 1 & -1 \end{bmatrix}.$$

B has no PSD completion here; if we transfer the positive off-diagonal weight on P to B , we can never get a PSD matrix. Therefore, we can see that A is exceptional.

5.3 Odd powers of exceptional copositive matrices

In this section, we would like to discuss our observation on odd powers of exceptional CoP matrices.

The motivation comes from the fact that CoP matrices are generalization of PSD matrices. In the PSD case, a polynomial $p(A)$ can preserve the positive semidefiniteness if it sends a positive number to a positive number, i.e. $p(t) \geq 0$ for $t \geq 0$. Thus, we wonder when does a polynomial $p(A)$ preserves copositivity.

It is clear that even powers of CoP matrices are PSD, so the even powers preserve copositivity. However, it is ambiguous that whether odd powers can preserve copositivity. Therefore, we wonder if odd powers preserve copositivity.

To better understand this question, we want to use computer program to generate random CoP matrices, raise them to odd powers, and then check copositivity of these odd powers with the CoP checking algorithm presented in Appendix. However, we note that it is only possible to generate random ordinary matrices, since they are the sums of PSD and sN matrices. Currently, there is no clear way of generating random exceptional matrices. Therefore, we only have a few number of known exceptional CoP matrices (presented in [15]) for our test.

In particular, we raised our random ordinary CoP matrices to the $3^{rd}, 5^{th}, 7^{th}$ powers, and we did this for size $n = 4, 5, 6, 7$, each with 100,000 random ordinary CoP matrices. All random ordinary CoP matrices we have tested are still CoP after raising them to the $3^{rd}, 5^{th}, 7^{th}$ powers. Additionally, we would like to indicate the limitation here, as counterexample may have a specific pattern that is hard to be randomly generated.

However, in exceptional CoP cases, the cubes are no longer CoP. We present the Horn matrix as an example below.

Example 5.3.1. Let A be the Horn Matrix. Then,

$$A = \begin{bmatrix} 1 & -1 & 1 & 1 & -1 \\ -1 & 1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & 1 & -1 \\ -1 & 1 & 1 & -1 & 1 \end{bmatrix}, A^3 = \begin{bmatrix} 13 & -11 & 5 & 5 & -11 \\ -11 & 13 & -11 & 5 & 5 \\ 5 & -11 & 13 & -11 & 5 \\ 5 & 5 & -11 & 13 & -11 \\ -11 & 5 & 5 & -11 & 13 \end{bmatrix}.$$

Let B be the principal submatrix of A^3 with the first three rows & columns, so $B = A[\{1, 2, 3\}]$. Then, B has eigenvalue $\lambda = -0.2560$ with eigenvector $v = \begin{bmatrix} 0.4586 \\ 0.7611 \\ 0.4586 \end{bmatrix}$. Thus, A^3 is not CoP by Kaplan's Theorem.

Therefore, we arrive at the following conjecture. If this conjecture is true, then taking the odd power of a CoP matrix would be a way to classify whether a CoP matrix is ordinary or exceptional.

Conjecture 5.3.2. Let $A \in M_n(\mathbb{R})$ be a CoP matrix. Then, any odd power of A is CoP if and only if A is ordinary.

5.4 3-by-3 Ordinary Decomposition

5.4.1 Decomposing a 3-by-3 CoP matrix

We give a 3-by-3 ordinary CoP decomposition method as below. From our prior discussion in Ch. 2.2, all 3-by-3 CoP matrices are ordinary. *Wlog*, we may take all diagonal entries to be 1's by diagonal equivalence. This follows from Prop. 2.1.1. Let $d, e, f \geq 0$, then we have 4 cases up to permutation similarity:

$$A_1 = \begin{bmatrix} 1 & d & f \\ d & 1 & e \\ f & e & 1 \end{bmatrix}; A_2 = \begin{bmatrix} 1 & -d & f \\ -d & 1 & e \\ f & e & 1 \end{bmatrix}; A_3 = \begin{bmatrix} 1 & -d & f \\ -d & 1 & -e \\ f & -e & 1 \end{bmatrix}; A_4 = \begin{bmatrix} 1 & -d & -f \\ -d & 1 & -e \\ -f & -e & 1 \end{bmatrix}.$$

For A_1 , an ordinary decomposition is give by

$$A_1 = \begin{bmatrix} 1 & d & f \\ d & 1 & e \\ f & e & 1 \end{bmatrix} = I_3 + \begin{bmatrix} 0 & d & f \\ d & 0 & e \\ f & e & 0 \end{bmatrix}.$$

For A_2 , since it is CoP, by Prop. 2.1.1, which stats $a_{ij} \geq -\sqrt{a_{ii}a_{jj}}$, we know the upper 2-by-2 submatrix of A_2 $\begin{bmatrix} 1 & -d \\ -d & 1 \end{bmatrix}$ is PSD. Thus, the following gives a valid ordinary decomposition for A_2 :

$$A_2 = \begin{bmatrix} 1 & -d & f \\ -d & 1 & e \\ f & e & 1 \end{bmatrix} = \begin{bmatrix} 1 & -d & 0 \\ -d & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 & f \\ 0 & 0 & e \\ f & e & 0 \end{bmatrix}.$$

For A_4 , it is already PSD. Therefore, we only need to work on A_3 .

We use the same idea as the naive decomposition in Sec. 5.2. Let $A = A_3 = \begin{bmatrix} 1 & -d & f \\ -d & 1 & -e \\ f & -e & 1 \end{bmatrix}$; $B = \begin{bmatrix} 1 & -d & 0 \\ -d & 1 & -e \\ 0 & -e & 1 \end{bmatrix}$. Let $B' = \begin{bmatrix} 1 & -d & f' \\ -d & 1 & -e \\ f' & -e & 1 \end{bmatrix}$, for $0 \leq f' \leq f$. If B is PSD, then we are done. Thus, we suppose that B is not PSD, and we want to make B' PSD by picking a suitable value f' between 0 and f .

Before we start to pick the f' , we would like to note that we never need to make a negative entry of a CoP matrix more negative to obtain an ordinary decomposition. In this case, we do not need to alter the $-d, -e$ entries. Since $d^2 \leq 1$, $\det\left(\begin{bmatrix} 1 & -d \\ -d & 1 \end{bmatrix}\right) \geq 0$. we need $\det(B') \geq 0$. Since $\det(B) = 1 - d^2 - e^2$, if we make $-d, -e$ more negative, then $\det(B)$ will be more negative. Therefore, we only need to adjust the positive entries, namely, f in B .

Consider B' . For B' to be PSD, we want $\det(B') > 0$.

Since $\det(B') = -f'^2 + 2def' - d^2 - e^2 + 1$, we solve the quadratic equation at $\det(B') = 0$, and the solution is

$$f' = de \pm \sqrt{(d-1)(d+1)(e-1)(e+1)}.$$

For easier notation, we let $f_1 = de - \sqrt{(d-1)(d+1)(e-1)(e+1)}$, $f_2 = de + \sqrt{(d-1)(d+1)(e-1)(e+1)}$, and we note that $f_1, f_2 > 0$.

Then, as f' increases from 0 to f , the determinant $\det(B')$ changes as the following:

1. When $0 < f' < f_2$, $\det(B') < 0$;
2. When $f_1 < f' < f_2$, $\det(B') > 0$;
3. When $f' > f_2$, $\det(B') < 0$.

5.4.2 Observations on eigenvalues and eigenvectors

In Kaplan's Theorem (Thm. [12]), the eigenvalue and eigenvector behavior of a copositive matrix seems interesting by itself. When we conduct computational experiment with 3-by-3 CoP matrices, we also notice some interesting behavior. Here, we want to remark some observations on eigenvalues and eigenvectors of 3-by-3 CoP matrices when we slowly increase f' from 0 to f as stated above (Sec. 5.4.1).

We use the same notation as Sec. 5.4.1, and let $B' = \begin{bmatrix} 1 & -d & f' \\ -d & 1 & -e \\ f' & -e & 1 \end{bmatrix}$. Let us denote the smallest eigenvalue of B' by λ_s and v_s . By our assumption, at $f = 0$, B' is not PSD. Thus, at $f = 0$, we have $\lambda_s < 0, v_s > 0$, and λ_s is the only negative eigenvalue of B' from the interlacing inequality.

If we trace λ_s and v_s as f' increases from 0 to f , we notice the following.

1. When $0 \leq f' < f_1$: $\lambda_s < 0$, and $v_s > 0 \Rightarrow B'$ is **not CoP**.
2. At $f' = f_1$: $\lambda_s = 0, v_s > 0 \Rightarrow B'$ is **PSD**.
3. At $f_1 < f' < f_2$, there is a certain point where v_s start to have mixed signs $\Rightarrow B'$ is **PSD**.
4. At $f' > f_2$: $\lambda_s < 0, v_s$ has mixed signs $\Rightarrow B'$ is **CoP, not PSD**.

We would like to further indicate some importance of these spectral related observation of ordinary CoP matrices. The inverse eigenvalue problem for symmetric

nonnegative (sN) matrices is unsolved, and ordinary CoP matrices can be decomposed into PSD + sN. If we take a sN matrix and add a PSD matrix to it, the eigenvalues never decreases. Therefore, it is valuable to study the spectrum of CoP matrices.

Chapter 6

Ordinary Copositive Matrices and their Graphs

Graph theory is also a common method to study matrices, and there are many interesting theories connecting the combinatorial aspect of graphs to matrices[cite]. In this chapter, we would like to talk about the graphs of ordinary CoP matrices.

6.1 Preliminaries

To begin our discussion about graph or ordinary graphs, we first give the following preliminaries in graph theory.

Definition 6.1.1. Given a symmetric matrix $A = (a_{ij}) \in M_n(\mathbb{R})$ and a graph G with n vertices, we say this matrix A *induces* the graph G if the following holds: if the off-diagonal entry $a_{ij}(i \neq j)$ is nonzero, then there is an edge between i and j ; if the off-diagonal entry $a_{ij}(i \neq j)$ is zero, then there no edge between i and j . We note that the diagonal entries of A can be any arbitrary numbers.

In this section, we use the notation $\mathcal{R}(G) = \{A \in M_n(\mathbb{R}) : G(A) = G\}$ to describe the set of all matrices that induce the graph G .

Definition 6.1.2. We call a graph G *ordinary* (in some literature, *SPN* [16]) if $A \in M_n(\mathbb{R})$ is ordinary for all $A \in \mathcal{R}(G) \cup \mathcal{C}_n$. In other words, a graph G is ordinary

if all copositive matrices associated with the graph G are ordinary.

The following lemma presents the main reason that people are interested in ordinary graph. Specifically, all subgraphs of an ordinary graph are ordinary.

Lemma 6.1.3. *If G is ordinary, then every subgraph of G is ordinary.*

Proof. This follows from Lem. 2.0.1. Since all principal submatrices of an ordinary CoP matrix is ordinary, all subgraphs of an ordinary graph are ordinary. \square

6.2 Properties Ordinary Graphs

In this section, we will discuss properties of ordinary graphs and some ordinary-preserving operation on ordinary graphs [16].

Lemma 6.2.1. *Let two ordinary graphs G_1 and G_2 be connected by a positive edge. Then, the resulting graph G is ordinary.*

Proof. Suppose $A_1 \in \mathcal{R}(G_1), A_2 \in \mathcal{R}(G_2)$ have ordinary decompositions: $A_1 = P_1 + S_1; A_2 = P_2 + S_2$. Wlog, let the new positive edge be added between the i, j vertices of the new graph G , and we denote the edges by k_{ij}, k_{ji} . Let K be the matrix that only contains k_{ij}, k_{ji} . Then, an ordinary decomposition of G is given by $P + S$, with $P = P_1 \otimes P_2, S = S_1 \otimes S_2 + K$. Therefore, G is ordinary. \square

Definition 6.2.2. Let $A \in M_p(\mathbb{R}), B \in M_q(\mathbb{R})$, and let A, B be partitioned as below:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix},$$

where $A_{11} \in M_{p-n}(\mathbb{R}), A_{12} \in M_{p-n,n}(\mathbb{R}), A_{21} \in M_{n,p-n}(\mathbb{R}), A_{22} \in M_n(\mathbb{R}); B_{11} \in M_n(\mathbb{R}), B_{12} \in M_{n,q-n}(\mathbb{R}), B_{21} \in M_{q-n,n}(\mathbb{R}), B_{22} \in M_{q-n}(\mathbb{R})$. Let $1 < n < p + q$.

Then, the n^{th} -subdirect sum of A and B is a matrix $C \in M_{p+q-n}$ such that

$$C = \begin{bmatrix} A_{11} & A_{12} & 0 \\ A_{21} & A_{22} + B_{11} & B_{12} \\ 0 & B_{21} & B_{22} \end{bmatrix}.$$

We give an example below to illustrate the 1st-subdirect sum.

Example 6.2.3. Let $A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$, $B = \begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}$. Then, the 1st-subdirect sum of A and B is

$$C = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 3 & 2 & 2 \\ 0 & 0 & 2 & 2 & 2 \\ 0 & 0 & 2 & 2 & 2 \end{bmatrix}.$$

Lemma 6.2.4. Let $G = G_1 \cup G_2$, where G_1, G_2 are ordinary graphs, and $G_1 \cap G_2$ is a single vertex. Then G is an ordinary graph if and only if G_1 and G_2 are ordinary graphs.

Proof. Again, suppose $A_1 \in \mathcal{R}(G_1)$, $A_2 \in \mathcal{R}(G_2)$ have ordinary decompositions: $A_1 = P_1 + S_1$; $A_2 = P_2 + S_2$. Wlog, let the i^{th} vertex, v_i be in both G_1 and G_2 . Then, all CoP matrices with graph G have ordinary decomposition with $P + S$, where P is the 1st-subdirect sum of P_1 and P_2 , and S is the 1st-subdirect sum of S_1 and S_2 . \square

Theorem 6.2.5. Let G_1, G_2 be ordinary graphs, and let $A_1 \in \mathcal{R}(G_1)$, $A_2 \in \mathcal{R}(G_2)$. Let B be the n^{th} -subdirect sum of A_1 and A_2 . Let G be the graph of B . Then, G is an ordinary graph.

Proof. We use the same idea as Lemma 6.2.4. Let A_1, A_2 have ordinary decompositions: $A_1 = P_1 + S_1$; $A_2 = P_2 + S_2$. Then, all CoP matrices with graph G have ordinary decomposition with $P + S$, where P is the n^{th} -subdirect sum of P_1 and P_2 , and S is the n^{th} -subdirect sum of S_1 and S_2 . \square

Definition 6.2.6. A *block* of G is a subgraph that has no cut vertex, and is maximal with respect to this property.

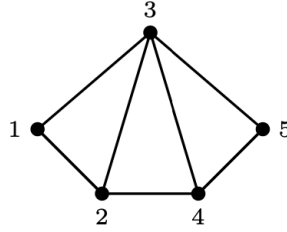


Figure 6.1: The Fan graph F_5 [13]

Lemma 6.2.7. *A graph G is ordinary if and only if every block of G is ordinary.*

A proof of this Lemma can be found in [13].

Lemma 6.2.8. *If we do an edge subdivision on an exceptional graph G and obtain G' , then G' is also exceptional.*

A proof of this Lemma can be found in [13].

6.3 Specific ordinary graphs

Lemma 6.3.1. *All trees are ordinary.*

Proof. Since each individual vertex of a tree can be treated as an 1-by-1 ordinary CoP matrix, this is a special case inherited from Lemma 6.2.7. \square

Lemma 6.3.2. *All cycles are ordinary.*

A proof of this Lemma can be found in [13].

6.4 Ordinary graphs on 5 vertices

Lemma 6.4.1. *The graph F_5 (shown in Fig.6.1), known as the fan graph, is an exceptional graph [13].*

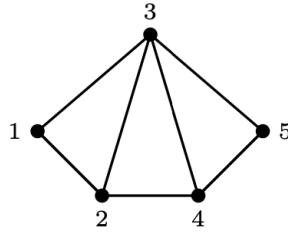


Figure 6.2: The exceptional graph CD_6 [13]

According to [13], ordinary graphs on 5 vertices are completely characterized as the following.

Theorem 6.4.2. *A graph on 5 vertices is an SPN graph if and only if it does not contain the fan F_5 [13].*

6.5 Ordinary graphs on 6 vertices

Lemma 6.5.1. *The graph CD_6 , shown in Fig., is an exceptional graph [13].*

Ordinary graphs on 6 vertices have not yet been fully characterized. In [13], the author wrongly thought that the graph T_6 , shown in Fig. 6.3 is ordinary. Later on in [14], Drury found a counterexample in matrix which shows that T_6 is not SPN.

In addition, we identify three boundary graphs with 6 vertices as shown in Fig. 6.4, Fig. 6.5, and Fig. 6.6. Namely, if we add an edge to these graphs, then the resulting graphs will be exceptional; if we take an edge away from these graph, then the resulting graphs will be ordinary. If we can settle down the three boundary graphs, then ordinary graphs on 6 vertices will be fully characterized.

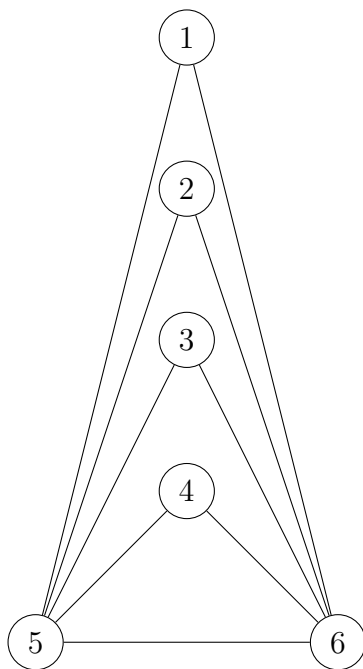


Figure 6.3: T_6 , an exceptional graph on 6 vertices [13]

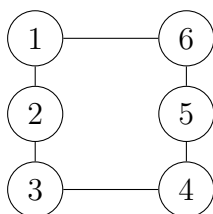


Figure 6.4: Boundary graph #1

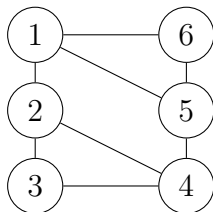


Figure 6.5: Boundary graph #2

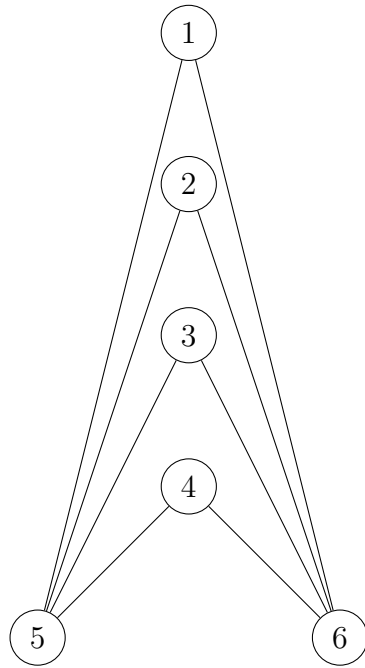


Figure 6.6: Boundary graph #3

Appendix

6.6 Python Codes for Checking Copositivity

In this section, we present the Python codes of checking copositivity for a given matrix with size $n = 5, 6, 7$. These programs are used and mentioned throughout our paper, and they are based on Kaplan's theorem in Sec. 3.3.

Also, we would like to explain the computational error (the parameter `com_err` in codes below). Since numerical calculations in Python give rise to error, it may be possible that a certain value should be exactly 0, but due to the computational error, is close to 0; for example, $10e-10$ instead of 0. Therefore, we include a computational error to serve as a lower/upper bound when determining whether a certain calculated value is positive/negative. It can be adjusted accordingly in the program.

The following import lines are needed for all codes:

```
#import lines
import math
import numpy as np
import scipy as sp
from numpy import linalg as lin
```

6.6.1 5-by-5 CoP Checking

```
#check whether a 5-by-5 matrix A is copositive or not
def Check_CoP(A): #Input: suppose A is a 5-by-5 matrix

    com_err = -10e-14 #compensate for the computational error

    #1st, check diagonal entries
    for i in range(5):
```

```

    if A[i,i]<0:
        #print("Not copositive at diagonal entries", A)
        return 1 #return 1 if not CoP

#2nd, check 2-by-2 principal submatrices
for i in range(4):
    for j in range(i+1,5):
        submatrix = A[[[i],[j]],[i,j]]
        eigval, eigvec = lin.eig(submatrix)
        for m in range(2):
            if eigval[m]<0:
                #print(eigval, eigvec)
                if eigvec[:,m][0]>0 and eigvec[:,m][1] >0:
                    #print(eigvec[:,m][1])
                    #print(eigval, eigvec)
                    print("Not copositive at 2-by-2, with: ",\
                        submatrix, eigval[m], eigvec[:,m])
                    return 1
                elif eigvec[:,m][0]<0 and eigvec[:,m][1] <0:
                    return 1

#3rd, check 3-by-3 principal submatrices
three_list = [ A[[[0],[1],[2]],[0,1,2]],\
                A[[[0],[1],[3]],[0,1,3]],\
                A[[[0],[1],[4]],[0,1,4]],\
                A[[[0],[2],[4]],[0,2,4]],\
                A[[[0],[2],[3]],[0,2,3]],\
                A[[[0],[3],[4]],[0,3,4]],\
                A[[[1],[2],[3]],[1,2,3]],\
                A[[[1],[2],[4]],[1,2,4]],\
                A[[[1],[3],[4]],[1,3,4]],\
                A[[[2],[3],[4]],[2,3,4]]]
for i in three_list:
    eigval, eigvec = lin.eig(i)
    for m in range(3):
        if eigval[m]<com_err:
            if eigvec[:,m][0]>com_err and \
                eigvec[:,m][1] >com_err and \
                eigvec[:,m][2] >com_err:
                print("Not copositive at 3-by-3, with: ", \
                    i, eigval[m], eigvec[:,m])
                return 1
            elif eigvec[:,m][0]<com_err and \
                eigvec[:,m][1]<com_err and \
                eigvec[:,m][2] <com_err:
                return 1

#4th, check 4-by-4 principal submatrices
four_list = [ A[[[0],[1],[2],[3]],[0,1,2,3]],\
                A[[[0],[1],[2],[4]],[0,1,2,4]],\
                A[[[0],[1],[3],[4]],[0,1,3,4]],\

```

```

        A[[[0],[2],[3],[4]],[0,2,3,4]],\
        A[[[1],[2],[3],[4]],[1,2,3,4]] ]
for i in four_list:
    eigval, eigvec = lin.eig(i)
    for m in range(4):
        if eigval[m]<com_err:
            if eigvec[:,m][0]>com_err and \
               eigvec[:,m][1]>com_err and \
               eigvec[:,m][2]>com_err and \
               eigvec[:,m][3]>0:
                print("Not copositive at 4-by-4, with: ", \
                      i, eigval[m], eigvec[:,m])
                return 1
            elif eigvec[:,m][0]<0 and eigvec[:,m][1]<0 \
                  and eigvec[:,m][2]<0 and eigvec[:,m][3]<0:
                return 1

#5th, check matrix A
eigval_A, eigvec_A = lin.eig(A)
for m in range(5):
    if eigval_A[m]<0:
        if eigvec_A[:,m][0]>0 and eigvec_A[:,m][1]>0 and \
           eigvec_A[:,m][2]>0 and eigvec_A[:,m][3]>0 and \
           eigvec_A[:,m][4]>0:
            print("Not copositive at 5-by-5, with: ", \
                  A, eigval_A[m], eigvec_A[:,m])
            return 1
        if eigvec_A[:,m][0]<0 and eigvec_A[:,m][1]<0 and \
           eigvec_A[:,m][2]<0 and eigvec_A[:,m][3]<0 and \
           eigvec_A[:,m][4] <0:
            return 1
print("It's CoP!")
return None

```

6.6.2 6-by-6 CoP Checking

```

#6x6 CoP Check REWRITE --- 8/11/2021
#check whether a 6-by-6 matrix is copositive or not

def Check_CoP_6(A): #suppose A is a 6-by-6 matrix

    com_err = -10e-14 #compensate the computational error

    #1st, check diagonal entries (and whether the big 6-by-6 is good
    )

    for i in range(6):
        if A[i,i]<0:
            print("Not copositive at diagonal entries", A)
            return 1 #return 1 if not CoP

```

```

eigval, eigvec = lin.eig(A)
for m in range(6):
    if eigval[m]<com_err:
        if (eigvec[:,m][0]>-0 and eigvec[:,m][1]>-0 and \
            eigvec[:,m][2]>-0 and eigvec[:,m][3]>-0 and \
            eigvec[:,m][4]>-0 and eigvec[:,m][5]>-0):
            print("A: Not copositive at 6-by-6", eigval[m], \
                eigvec[:,m])
            return 1
        elif (eigvec[:,m][0]<com_err and eigvec[:,m][1]<com_err \
            and eigvec[:,m][2]<com_err \
            and eigvec[:,m][3]<com_err \
            and eigvec[:,m][4]<com_err \
            and eigvec[:,m][5]<com_err): #check if they are
                                        the same signs
            print("B: Not copositive at 6-by-6", \
                eigval[m], eigvec[:,m])
            return 1

#2nd, check 2-by-2 principal submatrices
for i in range(5):
    for j in range(i+1,6):
        submatrix = A[[[i],[j]],[i,j]]
        eigval, eigvec = lin.eig(submatrix)
        for m in range(2):
            if eigval[m]<com_err:
                #print(eigval, eigvec)
                if (eigvec[:,m][0] * eigvec[:,m][1]) \
                    >com_err:
                    #check if they are the same signs
                    print("Not copositive at 2-by-2, with: ",\
                        submatrix, eigval[m], eigvec[:,m])
                    return 1

#3rd, check 3-by-3 principal submatrices
for i in range(4):
    for j in range(i+1,6):
        for k in range(j+1, 6):
            submatrix = A[[[i],[j],[k]],[i,j,k]]
            eigval, eigvec = lin.eig(submatrix)
            for m in range(3):
                if eigval[m]<com_err:
                    if (eigvec[:,m][0]>com_err and \
                        eigvec[:,m][1]>com_err and \
                        eigvec[:,m][2]>com_err) \
                    or (eigvec[:,m][0]<com_err and \
                        eigvec[:,m][1]<com_err and \
                        eigvec[:,m][2]<com_err): \
                        #check if they are the same signs
                        print("Not CoP at 3-by-3, with:",\

```

```

        submatrix, eigval[m], \
        eigvec[:,m],(i,j,k), A)
    return 1

#4th, check 4-by-4 principal submatrices
for i in range(3):
    for j in range(i+1,6):
        for k in range(j+1, 6):
            for f in range(k+1, 6):
                submatrix = A[[[i],[j],[k],[f]],[i,j,k,f]]
                eigval, eigvec = lin.eig(submatrix)
                for m in range(4):
                    if eigval[m]<com_err:
                        if eigval[m]>com_err:
                            if (eigvec[:,m][0]>com_err and \
                                eigvec[:,m][1]>com_err and \
                                eigvec[:,m][2]>com_err and \
                                eigvec[:,m][3]>com_err) \
                                or (eigvec[:,m][0]<com_err and \
                                    eigvec[:,m][1]<com_err and \
                                    eigvec[:,m][2]<com_err and \
                                    eigvec[:,m][3]<com_err): \
                                #check if they are the same signs
                                print("Not CoP at 4-by-4, with: ",
                                      submatrix, eigval[m], \
                                      eigvec[:,m])
                                return 1

#5th, check 5-by-5 principal submatrices
for i in range(2):
    for j in range(i+1,6):
        for k in range(j+1, 6):
            for f in range(k+1, 6):
                for g in range(f+1, 6):
                    submatrix = A[[[i],[j],[k],[f],[g]],[i,j,k,f,g], \
                                   [i,j,k,f,g]]
                    eigval, eigvec = lin.eig(submatrix)
                    for m in range(5):
                        if eigval[m]<com_err:
                            if (eigvec[:,m][0]>com_err \
                                and eigvec[:,m][1]>com_err \
                                and eigvec[:,m][2]>com_err \
                                and eigvec[:,m][3]>com_err \
                                and eigvec[:,m][4]>com_err) \
                                or (eigvec[:,m][0]<com_err and \
                                    eigvec[:,m][1]<com_err and \
                                    eigvec[:,m][2]<com_err and \
                                    eigvec[:,m][3]<com_err and \
                                    eigvec[:,m][4]<com_err):
                                #check if same signs
                                print("Not CoP 5-by-5,w/", \
                                      submatrix, eigval[m], \

```

```

                                eigvec[:,m])
                                return 1

print("It's CoP!")
return None

```

6.6.3 7-by-7 CoP Checking

```

def Check_CoP_7(A): #suppose A is a 7-by-7 matrix
    com_err = -10e-14
    #1st, check diagonal entries (and whether the 7-by-7 is good)
    for i in range(7):
        if A[i,i]<0:
            print("Not copositive at diagonal entries", A)
            return 1 #return 1 if not CoP
    eigval, eigvec = lin.eig(A)

    #1st, check the diagonal entries
    for m in range(7):
        if eigval[m]<com_err:
            if (eigvec[:,m][0]>com_err \
                and eigvec[:,m][1]>com_err \
                and eigvec[:,m][2]>com_err \
                and eigvec[:,m][3]>com_err \
                and eigvec[:,m][4]>com_err \
                and eigvec[:,m][5]>com_err \
                and eigvec[:,m][6]>com_err):
                print("Not copositive at 7-by-7", \
                    eigval[m], eigvec[:,m])
                return 1
            elif (eigvec[:,m][0]<com_err \
                  and eigvec[:,m][1]<com_err \
                  and eigvec[:,m][2]<com_err \
                  and eigvec[:,m][3]<com_err \
                  and eigvec[:,m][4]<com_err \
                  and eigvec[:,m][5]<com_err \
                  and eigvec[:,m][6]<com_err):
                #check if they are the same signs
                print("Not copositive at 7-by-7", \
                    eigval[m], eigvec[:,m])
                return 1

    #2nd, check 2-by-2 principal submatrices
    for i in range(6):
        for j in range(i+1,7):
            submatrix = A[[[i],[j]],[i,j]]
            eigval, eigvec = lin.eig(submatrix)
            for m in range(2):
                if eigval[m]<com_err:
                    #print(eigval, eigvec)

```

```

        if (eigvec[:,m][0] * \
            eigvec[:,m][1])>com_err:
            #check if they are the same signs
            print("Not CoP at 2-by-2, w/: ",\
                submatrix, eigval[m], eigvec[:,m])
            return 1

#3rd, check 3-by-3 principal submatrices
for i in range(5):
    for j in range(i+1,7):
        for k in range(j+1, 7):
            submatrix = A[[[i],[j],[k]],[i,j,k]]
            eigval, eigvec = lin.eig(submatrix)
            for m in range(3):
                if eigval[m]<com_err:
                    if (eigvec[:,m][0]>com_err \
                        and eigvec[:,m][1]>com_err \
                        and eigvec[:,m][2]>com_err) \
                    or (eigvec[:,m][0]<com_err \
                        and eigvec[:,m][1]<com_err \
                        and eigvec[:,m][2]<com_err):
                        #check if they are the same signs
                        print("Not CoP at 3-by-3, w/: ",\
                            submatrix,eigval[m],eigvec[:,m],\
                                (i,j,k), A)
                        return 1

#4th, check 4-by-4 principal submatrices
for i in range(4):
    for j in range(i+1,7):
        for k in range(j+1, 7):
            for f in range(k+1, 7):
                submatrix = A[[[i],[j],[k],[f]],[i,j,k,f]]
                eigval, eigvec = lin.eig(submatrix)
                for m in range(4):
                    if eigval[m]<com_err:
                        if (eigvec[:,m][0]>com_err \
                            and eigvec[:,m][1]>com_err \
                            and eigvec[:,m][2]>com_err \
                            and eigvec[:,m][3]>com_err) \
                        or (eigvec[:,m][0]<com_err \
                            and eigvec[:,m][1]<com_err \
                            and eigvec[:,m][2]<com_err \
                            and eigvec[:,m][3]<com_err):
                            #check if they are the same signs
                            print("Not CoP at 4-by-4, w/",\
                                submatrix,eigval[m],\
                                    eigvec[:,m])
                            return 1

```



```

#5th, check 5-by-5 principal submatrices
for i in range(3):
    for j in range(i+1,7):
        for k in range(j+1, 7):
            for f in range(k+1, 7):
                for g in range(f+1, 7):
                    submatrix = A[[[i],[j],[k],[f],[g]],\
                                   [i,j,k,f,g]]
                    eigval, eigvec = lin.eig(submatrix)
                    for m in range(5):
                        if eigval[m]<com_err:
                            if eigvec[:,m]>com_err \
                                and eigvec[:,m][1]>com_err \
                                and eigvec[:,m][2]>com_err \
                                and eigvec[:,m][3]>com_err \
                                and eigvec[:,m][4]>com_err) \
                                or (eigvec[:,m][0]<com_err \
                                    and eigvec[:,m][1]<com_err \
                                    and eigvec[:,m][2]<com_err \
                                    and eigvec[:,m][3]<com_err \
                                    and eigvec[:,m][4]<com_err):
                                    #check if same signs
                                    print("Not CoP at 5-by-5 w/", \
                                          submatrix,eigval[m],\
                                          eigvec[:,m])
                                    return 1

#6th, check 6-by-6 principal submatrices
for i in range(2):
    for j in range(i+1,7):
        for k in range(j+1, 7):
            for f in range(k+1, 7):
                for g in range(f+1, 7):
                    for h in range(g+1, 7):
                        submatrix = A[[[i],[j],[k],[f],[g],\
                                         [h]], [i,j,k,f,g,h]]
                        eigval, eigvec = lin.eig(submatrix)
                        for m in range(6):
                            if eigval[m]<com_err:
                                if (eigvec[:,m][0]>com_err \
                                    and eigvec[:,m][1]>com_err \
                                    and eigvec[:,m][2]>com_err \
                                    and eigvec[:,m][3]>com_err \
                                    and eigvec[:,m][4]>com_err \
                                    and eigvec[:,m][5]>com_err) \
                                    or (eigvec[:,m][0]<com_err \
                                        and eigvec[:,m][1]<com_err \
                                        and eigvec[:,m][2]<com_err \
                                        and eigvec[:,m][3]<com_err \
                                        and eigvec[:,m][4]<com_err \
                                        and eigvec[:,m][5]<com_err):

```

```
print("It's CoP!")  
return None  
  
#check if same signs  
print("Not CoP 6-by-6 w/", \  
      submatrix, eigval[m],\  
      eigvec[:,m])  
return 1
```