On Completely Positive Matrices

by

Torkel A. Haufmann

MASTER'S THESIS

For the degree

Master of Applied mathematics and mechanics

(Master of Science)

Det matematisk-naturvitenskapelige fakultet
Universitetet i Oslo

May 2011

Faculty of Mathematics and Natural Sciences
University of Oslo

Preface

It is a privilege to write a master's thesis, and certainly it is a unique experience. Before we begin, then, I wish to thank my advisor Geir Dahl for providing guidance and help throughout the process, and my girlfriend Mina Storhaug for suffering my ever-distracted company through the year.

Contents

Chapter 1

Introduction and Preliminaries

1.1 Introduction

The topic of this master's thesis is the notion of a *completely positive matrix*, which is a matrix that can be decomposed as

$$A = BB^T,$$

where B is an elementwise nonnegative matrix. Completely positive matrices have arisen in some situations in economic modelling and appear to have some applications in statistics, and they are also the dual cone of the cone of copositive matrices, which has been studied some in connection with quadratic optimisation (see as an example [6]). In this thesis, however, we are interested in the completely positive matrices as a theoretical object in and of themselves.

It is immediately apparent that a completely positive matrix is symmetric and nonnegative, and it is also clear that it must be positive semidefinite. Positive semidefinite and nonnegative matrices are known as doubly nonnegative matrices, and form a convex cone. The completely positive matrices also form a cone, but it turns out to be strictly contained in the cone of doubly nonnegative matrices.

The main problem concerning completely positive matrices is that it is not known how to test for membership in the completely positive cone. Several sufficient results are known, but the literature concerning necessary criteria seems a lot sparser. In this thesis we set before us three tasks:

1. We will give an overview of some central results concerning completely positive matrices.

2. The nonnegative diagonally dominant matrices are known to be completely positive. We will consider whether this is a piece of information from which we can learn more.

3. We will study some algorithms that attempt to determine whether or not a doubly nonnegative matrix is completely positive.

The first item will be the topic of Chapter 2, the second will be dealt with in Chapter 3 while the third will be the topic of Chapters 4 and 5. As such, Chapter 2 will contain theorems I have found elsewhere, while Chapters 3-5 will mainly be my own work unless otherwise noted (it seems like most research on completely positive matrices follows different paths than we will endeavour to do in this thesis).

Before we can begin with these three tasks, however, we wish to make clear the theory we are building on and the notation we use.

Nearly all the results in this first chapter will be known to any reader (In particular the first few sections can be skimmed, they only serve to establish notation, but nothing in this chapter is very advanced), and most can be found in any book on elementary finite-dimensional linear algebra, like [11], though we also use some results concerning convexity. The book [3], which is specifically about complete positivity, also covers much of the material, for the same reason we do – the definitions in this chapter are useful in the study of completely positive matrices. We obviously cannot restate all the theory we build on from first principles, so we simply provide definitions and theorems where warranted, and assume familiarity with some notions we do not define here[1], because they are not so much in the core of our theory. We will usually not go into proofs in this chapter.

Except where otherwise noted, the material in Sections 1.2-1.5 and 1.8 can be found in [3], while the material in Sections 1.6 and 1.7 can be found in [9]. Section 1.9 references [16], and Section 1.10 references [17].

1.2 Numbers, vectors and matrices

As is usual, we denote the real numbers by $\mathbb{R}$ and the natural numbers by $\mathbb{N}$. We will never have need for complex numbers in this thesis, so all matrices, matrix spaces, vectors, vector spaces and numbers will be implicitly assumed to be real unless otherwise stated. We also assume all vectors are column vectors unless otherwise stated. If S is a finite set, we let $|S|$ denote the size, or number of elements, of the set (we will not need to talk about the size of infinite sets).

For any positive integer n we let $\mathbb{R}^n$ denote the vector space of real column vectors of size n. Vectors will usually be denoted by boldface letters, like so: $\mathbf{x}$. The elements of some $\mathbf{x} \in \mathbb{R}^n$ will be referred to as $x_1, x_2, \ldots, x_n$. We will have occasion to refer to the *nonnegative orthant* of $\mathbb{R}^n$, denoted by $\mathbb{R}^n_+$, consisting of those vectors that are elementwise nonnegative, i. e.:

$$\mathbb{R}^n_+ = \{\mathbf{x} \in \mathbb{R}^n : x_i \geq 0 \text{ for } i = 1, \ldots, n\}.$$

If $\mathbf{x} \in \mathbb{R}^n_+$, we say that $\mathbf{x}$ is nonnegative, and we write $\mathbf{x} \geq \mathbf{0}$[2].

A set of vectors $\{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ in some vector space is said to be *linearly independent* if the vector equation

$$\mathbf{0} = a_1\mathbf{v}_1 + \ldots + a_k\mathbf{v}_k$$

[1] Particular examples include linear vector spaces, subspaces, convergence, gradients, convex functions and so forth.

[2] By a boldface zero, we mean a vector of all zeroes that has the appropriate dimension. The dimension is usually clear from context, so for notational simplicity we do not resolve the ambiguity of $\mathbf{0}$ by adding subscripts or the like.

with $a_1, \ldots, a_k$ appropriate scalars only has the solution $a_1 = \ldots = a_k = 0$. A combination of vectors of this kind is known as a *linear combination*. If V is some linear vector space and S a set of vectors in V such that all vectors in V can be written as a linear combination of vectors in S, we say that S is a *spanning set* for V. The set obtained from S by taking all possible linear combinations of vectors in S is denoted $\operatorname{Span} S$. A spanning set that is also linearly independent is known as a basis. The number of vectors needed in a basis for a space is known as its *dimension*, is independent of the basis chosen, and it is known that $\mathbb{R}^n$ has dimension n.

Beyond $\mathbf{0}$ we will also use the notation $\mathbf{1}_n$ for a vector in $\mathbb{R}^n$ whose elements are all ones, and we will denote the standard basis in the same space by $\mathbf{e}_i$ for $i = 1, \ldots, n$, that is, $\mathbf{e}_i$ is a vector consisting of all zeroes, except for position i, where there is a 1.

For positive integers m, n we let $\mathbb{R}^{m\times n}$ denote the space of $m \times n$ real matrices. If m or n are 1 then $\mathbb{R}^{m\times n}$ is a space of row or column vectors, respectively. The space $\mathbb{R}^{m\times n}_+$ is defined in the obvious way as the nonnegative orthant of $\mathbb{R}^{m\times n}$. Matrices will be denoted by capital letters, like so: A. We denote the (i, j)-th element, the element in row i and column j, of a matrix A by a_{ij}, or occasionally A_{ij}. If necessary, for instance when referring to an element like the one in position $(i + 1, i + 2)$ of a matrix A, we will use a comma, so we write $a_{i+1,i+2}$ as opposed to a_{i+1i+2} (which would be ambiguous). $\mathbb{R}^{m\times n}$ has dimension mn. The *rank* of a matrix is defined as the largest number of columns in it we can choose while forming a linearly independent set.

The *transpose* of a matrix A will be denoted by A^T. If $A = A^T$ then A is called a *symmetric* matrix. The set of symmetric nonnegative $n \times n$ matrices will be denoted by $\mathcal{S}^n_+$. If $a_{ij} \geq 0$ for all i, j we say that A is *nonnegative*, writing $A \geq O$[3].

We will assume the common definition of matrix multiplication, and multiplication of a matrix with a vector, where an $m \times n$-matrix transforms an n-dimensional vector into an m-dimensional one. When writing A^k for some k, we mean the product $A^k = AAA \ldots A$ with k matrices in total – note that this is only defined if A is square.

The *Hadamard product*, denoted by a $\circ$, is defined by $(A \circ B)_{i,j} = A_{i,j}B_{i,j}$, that is, componentwise multiplication. If A is a matrix, we let $A^{(k)}$ denote the k-th Hadamard power of A, for $k \geq 1$.

1.3 Special types of matrices

The elements of a square matrix A in the positions (i, i) for some i are known as the *diagonal elements* of the matrix. A square matrix D is called a diagonal matrix if the only nonzero elements of D are the diagonal elements. If each of the diagonal elements is strictly positive, D is known as a positive diagonal matrix (Note that $D \not> 0$, since all off-diagonal elements are 0). If we let $d_1, \ldots, d_n$ be the diagonal elements of D, we sometimes write $D = \operatorname{diag}(d_1, \ldots, d_n)$. A special

[3]This is another zero-related abuse of notation, where we let O be the matrix of all zeroes of appropriate dimension. Again, usually the context makes it clear which zero matrix is meant, so we do not bother resolving this ambiguity.

positive diagonal matrix is the identity matrix $I = \text{diag}(1, 1, \ldots, 1)$[4]. This has the property that $IA = AI = A$ for A and I of appropriate dimension.

A special kind of matrix is the *diagonally dominant* matrix. A matrix $A = [a_{ij}]_{i,j=1}^{n}$ is diagonally dominant if the following holds:

$$a_{ii} \geq \sum_{k=1, k\neq i}^{n} a_{ij} \text{ for } i = 1, \ldots, n.$$

We will return to diagonally dominant matrices several times throughout this thesis.

If P is a square $(0, 1)$-matrix with exactly one nonzero element (which is consequently 1) in each row and in each column, we call P a permutation matrix. Multiplying a matrix from the left by P corresponds to permuting the order of the rows in said matrix, while multiplying from the right permutes the columns. If we multiply from the left by P and from the right by P^T we get a matrix in which the columns and rows have been permuted in the same way.

If a matrix has the property that all the values above the diagonal are zero, it will be called *lower triangular*. We define *upper triangular* in the obvious way, and if a matrix is either we say it is triangular.

To any square matrix A we associate a matrix $M(A)$ of the same dimensions called the *comparison matrix*. It is defined by setting its (i, j)-th element to be $-|A_{ij}|$ if $i \neq j$ and $|A_{ij}|$ if $i = j$. It can be shown that if a comparison matrix is positive semidefinite (see section 1.5) there exists a positive diagonal matrix D such that $DM(A)D$ is diagonally dominant. We shall have use for this result in Chapter 2.

1.4 Euclidean vector spaces and norms

If V is an n-dimensional vector space with an inner product denoted by $\langle \cdot, \cdot \rangle$ and defined by

$$\text{for } \mathbf{v}, \mathbf{w} \in V, \langle \mathbf{v}, \mathbf{w} \rangle = v_1 w_1 + v_2 w_2 + \ldots + v_n w_n,$$

we say it is a *Euclidean* vector space. The inner product induces the *norm* of any $\mathbf{v} \in V$,

$$\|\mathbf{v}\| = \sqrt{\langle \mathbf{v}, \mathbf{v} \rangle} = \sqrt{\sum_{i=1}^{n} x_i^2}.$$

Using this, we can also talk about the angle θ between two vectors $\mathbf{v}, \mathbf{w} \in V$:

$$\theta = \cos^{-1} \frac{\langle \mathbf{v}, \mathbf{w} \rangle}{\|\mathbf{v}\| \|\mathbf{w}\|}.$$

Thus far we have assumed the elements of a Euclidean vector space are column vectors (basically, that we are in $\mathbb{R}^n$) – we will also be using matrix spaces. In that case we can write the inner product somewhat more succinctly, though it is still the same idea of adding up the elements that are in the "same positions"

[4] We will, as with the zero matrix, rarely bother to specify the size of I – it will be clear from context.

multiplied together. The *trace* of a matrix A, denoted $\operatorname{tr} A$, is the sum of its diagonal elements. We see that if $A, B \in \mathbb{R}^{n\times m}$,

$$\langle A, B\rangle = \operatorname{tr}(AB^T)$$

defines an inner product satisfying the same properties as the one above. In both cases, we have a Euclidean vector space. The norm induced by this inner product, called the *Frobenius* norm, is denoted by $\|\cdot\|_F$. We note also that elements in $\mathbb{R}^{n\times m}$ can, as vectors, as well be considered to be in $\mathbb{R}^{nm}$ – this is a property we will make use of later.

Using the definitions of angle and norms we can define *orthonormal* sets of vectors. If $\{\mathbf{v}_1, \dots \mathbf{v}_k\}$ is a set of vectors such that for all i, $\|\mathbf{v}_i\| = 1$, and for all i, j with $i \neq j$, $\langle \mathbf{v}_i, \mathbf{v}_j\rangle = 0$, then we say that the vectors are orthonormal.

We will have some cause to talk about isometries. A *linear isometry* is a map from one Euclidean vector space to another which preserves angles and vector lengths. It is a fact that any linear isometry (on a finite-dimensional space) can be represented by an orthogonal matrix U, that is a matrix such that $UU^T = U^TU = I$ (this is equivalent to the columns and rows of U being sets of orthonormal vectors). For us the importance of these isometries is that they preserve inner products.

If $\mathbf{v}_1, \dots, \mathbf{v}_n$ is some set of vectors in a Euclidean vector space, we define the *Gram matrix* $A = \operatorname{Gram}(\mathbf{v}_1, \dots, \mathbf{v}_n)$ by $A_{ij} = \langle \mathbf{v}_i, \mathbf{v}_n\rangle$. Note that in particular, a Gram matrix is symmetric, and if the greatest angle between any two vectors is at most $\pi/2$, it is also nonnegative.

Going back to the norm, we will have some use for the so-called p-norms, namely the 1-norm and the ∞-norm. They are defined for any vector $\mathbf{x} \in \mathbb{R}^n$ as follows:

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}. \tag{1.1}$$

The uniform norm is obtained by letting p tend to infinity, and we often simply say that we let $p = \infty$. It is defined by

$$\|\mathbf{x}\|_\infty = \max\{|x_i| : i = 1, \dots, n\}. \tag{1.2}$$

To make clear what it does we will usually say $\|\cdot\|_{max}$ instead of $\|\cdot\|_\infty$, as there is no difference in our finite-dimensional spaces. Matrix spaces are also vector spaces, and when we need norms on matrix spaces, we will often use these vector norms basically by pretending the matrix is a column vector obtained by stacking its columns on top of each other.

In any finite-dimensional space all norms are *equivalent*, which means (for our purposes) that they give convergence in the same situations, so when we are discussing convergence later we can choose norms only according to which is easier to compute.

1.5 Positive semidefinite matrices

We will be using positive semidefinite matrices a good deal. In this section, we define them and note some basic properties. We will not have need for all of these, but there is no harm in stating them all – the verbosity of the theorem

in this section does illustrate that the class of positive semidefinite matrices is well understood, as opposed to the class we are studying.

In many applications, quadratic forms come up. A *quadratic form* is an expression of the form $\mathbf{x}^T A\mathbf{x}$ for some symmetric matrix A, and it is obviously a polynomial in $x_1, x_2, \ldots, x_n$. Often, one needs to know when a quadratic form takes nonnegative values. We say that a matrix A such that the associated quadratic form is nonnegative for all $\mathbf{x}$ is *positive semidefinite*. There are a number of different characterisations of positive semidefiniteness, but we will only use a few, so we do not bother mentioning the others here.

Theorem 1.5.1. *Let A be an $n \times n$ symmetric matrix. Then the following are equivalent:*

1. *A is positive semidefinite.*
2. *Any eigenvalue of A is nonnegative.*
3. *There exists a lower triangular $n \times n$-matrix L such that $A = LL^T$.*
4. *There exists some $n \times k$-matrix B such that $A = BB^T$.*
5. *There exists a k-dimensional Euclidean vector space V and vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n \in V$ such that $A = \mathrm{Gram}(\mathbf{v}_1, \ldots, \mathbf{v}_n)$.*
6. *There exist k vectors $\mathbf{b}_1, \ldots, \mathbf{b}_k \in \mathbb{R}^n$ such that $A = \sum_{i=1}^{k} \mathbf{b}_i \mathbf{b}_i^T$.*

It is a fact that in the above results, k is equal to the rank of A.

The set of positive semidefinite $n \times n$ matrices will be referred to as $\mathcal{PSD}_n$. For us, the three last characterisations will be the most useful. The following properties are also well-known – we reference [13] for a particular source. They are all obvious to see using the representations in Theorem 1.5.1.

Theorem 1.5.2. *If A is positive semidefinite and of order n, the following holds.*

1. *$|a_{ij}| \leq \frac{1}{2}(a_{ii} + a_{jj})$ for $i = 1, \ldots, n-1, j = i+1, \ldots, n$.*
2. *$|a_{ij}| \leq \sqrt{a_{ii} a_{jj}}$ for $i = 1, \ldots, n-1, j = i+1, \ldots, n$.*
3. *$\max_{i,j} |a_{ij}| = \max_i a_{ii}$.*
4. *$a_{ii} = 0 \Rightarrow a_{ij} = a_{ji} = 0$ for any fixed i with $j = 1, \ldots, n$.*

In time, we will see that positive semidefinite matrices are an important superset of the set of completely positive matrices, the topic of this thesis. For now, we make note that there exists an effective algorithm for verifying positive semidefiniteness, namely *semi-Cholesky factorisation*, see for example [13] again, although this is found in most texts dealing at all with numerical linear algebra. This obtains a decomposition of the matrix into a lower triangular matrix (item 3 in Theorem 1.5.1), proving positive semidefiniteness. It will only fail to do so if the matrix is not positive semidefinite. The algorithm is well known, so we do not reproduce it here. It runs in $O(n^3)$ time – we will define what this is in Section 1.10.

1.6 Convexity, cones and dimension

Throughout this section, V will be assumed to be some finite dimensional Euclidean space. We will use $\mathbb{R}^n$ to make the results easier to read, but the notions in this section also hold for $\mathbb{R}^{n\times m}$, and it is that space we shall apply them to later.

A set $C \subseteq V$ is *convex* if $\mathbf{x}, \mathbf{y} \in C$ implies that for any $0 \leq \lambda \leq 1$, $\lambda\mathbf{x} + (1-\lambda)\mathbf{y} \in C$. It is reasonably easy to see that this is the same as requiring, for any set $\mathbf{x}_1, \ldots, \mathbf{x}_m \in C$, $\sum_{i=1}^m \lambda_i \mathbf{x}_i \in C$ if $\sum_{i=1}^m \lambda_i = 1$ and $\lambda_i \geq 0$ for all i. A set C is a *cone* if, for all $\mathbf{x} \in C$, $a\mathbf{x} \in C$ if $a \geq 0$. A set C is then a *convex cone* if for any $a, b \in \mathbb{R}^+$ and $\mathbf{x}, \mathbf{y} \in C$ we know that $a\mathbf{x} + b\mathbf{y} \in C$. The only convex sets we will be considering are convex cones. We will never use non-convex cones, so sometimes we will simply say cone instead of convex cone.

Proposition 1.6.1 (Convex cones)**.** *Let S be a set and let $\mathbf{x}_1, \ldots, \mathbf{x}_m \in S$. Then S is a convex cone if and only if for any $\lambda_1, \ldots, \lambda_m \in \mathbb{R}_+$ we have that*

$$\sum_{i=1}^m \lambda_i \mathbf{x}_i \in S.$$

Further, for any finite set $S = \{\mathbf{x}_i\}_{i=1}^m \subseteq \mathbb{R}^n$ we can form the set

$$C = \left\{ \sum_{i=1}^m \lambda_i \mathbf{x}_i : \lambda_1, \ldots, \lambda_m \in \mathbb{R}_+ \right\}.$$

The set C is the smallest convex cone containing S, and we will denote it by $\operatorname{cone} S$.

The latter part of this proposition will prove useful later. The big advantage it affords is when S is explicitly known, in which case Proposition 1.6.1 says that we often only have to prove properties for the elements in S as opposed to any element. The set S is referred to as the *generating set* of the cone $\operatorname{cone} S$, and the elements of S are referred to as *generators*.

A cone is often talked about in terms of its *extreme rays*. An extreme ray of a cone C is a set of the form $\{\lambda\mathbf{x} : \lambda \in \mathbb{R}_+, \mathbf{x} \in C\}$ with the property that if $\mathbf{x} = \frac{1}{2}(\mathbf{y} + \mathbf{z})$ with $\mathbf{y}, \mathbf{x} \in C$ then both $\mathbf{y}$ and $\mathbf{z}$ are nonnegative multiples of $\mathbf{x}$, that is, they are themselves in the set.

An example of an important cone is the one of *doubly nonnegative matrices*, that is, matrices that are both nonnegative and positive semidefinite. The cone of doubly nonnegative $n \times n$ matrices is denoted by $\mathcal{DNN}_n$. It is obviously a cone, and it is contained in $\mathcal{S}_+^n$ (which is also a cone), since all positive semidefinite matrices are symmetric.

For any convex set in $\mathbb{R}^n$ we can define a notion of dimension, but first we need the notion of *affine independence*. In a Euclidean vector space V a set of vectors $\mathbf{v}_1, \ldots, \mathbf{v}_k$ is affinely independent if, for any set of scalars $\lambda_1, \ldots, \lambda_k \in \mathbb{R}$, $\sum_{i=1}^k \lambda_i \mathbf{v}_i = \mathbf{0}$ and $\sum_{i=1}^k \lambda_i = 0$ imply that $\lambda_1 = \ldots = \lambda_k = 0$. It can be proved that the condition of affine independence is equivalent to requiring that the vectors $\mathbf{v}_2 - \mathbf{v}_1, \ldots, \mathbf{v}_k - \mathbf{v}_1$ are linearly independent.

Definition 1.6.2 (Dimension of a subset of $\mathbb{R}^n$)**.** *The dimension of a set $S \subseteq \mathbb{R}^n$ is the maximal number of affinely independent points in S minus 1.*

This definition of dimension coincides with the standard definition of dimension with regards to linear subspaces, and it should be easy to see that in a convex cone it is equivalent to the maximal number of linearly independent vectors we can find (Take some linearly independent points in the cone and adjoin to them the origin, if necessary after translating the cone).

1.7 Hyperplanes, halfspaces and polyhedra

The results in this section remain general, but it is easier to describe them in $\mathbb{R}^n$, and so we will do that[5]. A *hyperplane* in $\mathbb{R}^n$ is a set of the form $\{\mathbf{x} \in \mathbb{R}^n : \langle \mathbf{x}, \mathbf{y} \rangle = a\}$ for some $\mathbf{y} \in \mathbb{R}^n, a \in \mathbb{R}$. A *halfspace* is a set of the form $\{\mathbf{x} \in \mathbb{R}^n : \langle \mathbf{x}, \mathbf{y} \rangle \leq a\}$ for some $\mathbf{y} \in \mathbb{R}^n, a \in \mathbb{R}$. Any halfspace is a convex set, and a set which is a finite intersection of halfspaces is known as a *polyhedron*, which clearly is a convex set. In particular, a (convex) cone C which is such that $C = \operatorname{cone} S$ for some finite set S, is a polyhedron, and is known as a *polyhedral cone*. This is the kind of cone we will mostly be working with.

It can be shown that a set is a polyhedron if and only if it is defined by a set of linear inequalities, and so for any polyhedron P we can find a matrix A and a vector $\mathbf{b}$ such that

$$P = \{\mathbf{x} \in \mathbb{R}^n : A\mathbf{x} \leq \mathbf{b}\}.$$

Note that the representation need not be unique: Some of the rows in A may be redundant. For instance, consider the following.

$$A = \begin{bmatrix} 1 & 1 \\ 2 & 2 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 1 \\ 10 \end{bmatrix}.$$

The second inequality here clearly follows from the first, and so it is redundant. A system of inequalities that contains no redundant inequalities is said to be *irredundant*.

Further, we define a *face* of a polyhedron P as a subset F with the following property: If $x \in P$ and we can write x as a convex combination of two points $x_1, x_2 \in F$ then it follows that $x \in F$. Any convex set is a face of itself, being convex, and so is the empty set by convention, but most sets will contain nontrivial faces – an example of (usually) nontrivial faces are the extreme rays we defined in Section 1.6. A *facet* is a nontrivial face of maximal dimension, that is, if P is a polyhedron and F is a face, then F is a facet if and only if $\dim F = \dim P - 1$.

The reason we care about facets is that they help us find minimal linear systems defining a polyhedron, which is to say systems without redundant inequalities. Assume P is a polyhedron, in which case it is described by some number of inequalities. We assume each of them are non-redundant, which in particular means that each of them are sometimes fulfilled with equality. The set for which there is equality in an inequality is known as an *exposed face*, and it is easy to see that it is actually a face. It is shown, for example in [14], that

[5] In particular, it is easier to describe a linear inequality system in $\mathbb{R}^n$, because we can simply say $A\mathbf{x} \leq \mathbf{b}$. If we wished to study matrix spaces we would have to introduce more cumbersome notation here. Later in the thesis, though, we will always be applying these results to matrix spaces in practice.

for a polyhedral cone all faces are exposed faces. An inequality whose related exposed face is a facet is known as a *facet-defining inequality.*

The following three theorems cover what we need to know about the situation, and they can be found in [14]. In all three, we let $P = \{A\mathbf{x} \leq \mathbf{b} : \mathbf{x} \in \mathbb{R}^n\}$ be a polyhedron in the space $\mathbb{R}^n$.

Theorem 1.7.1. *If P has full dimension in the space ($\dim P = n$) and the system $A\mathbf{x} \leq \mathbf{b}$ is irredundant, then every inequality in the system defines a facet.*

Theorem 1.7.2. *If P has full dimension and the system $A\mathbf{x} \leq \mathbf{b}$ is irredundant, then it is the unique minimal representation of P as the solution set of a system of linear inequalities (up to multiplying any inequality by a scalar).*

Theorem 1.7.3. *If P has full dimension the system $A\mathbf{x} \leq \mathbf{b}$ is irredundant if and only if, for any two inequalities from the system $\alpha_i\mathbf{x} \leq b_i$ and $\alpha_j\mathbf{x} \leq b_j$ there exists an element $\mathbf{x}_0 \in P$ such that one inequality holds with equality and the other does not.*

Together Theorems 1.7.1-1.7.3 will let us find minimal defining inequality systems in Chapter 3.

Now we define a special kind of convex cone, which will illustrate why facets are useful.

Definition 1.7.4 (Simplicial cones)**.** *A* simplicial cone *is a finitely generated cone $C = \operatorname{cone} S$ such that S is a set of linearly independent vectors. Clearly we must have $\dim C = |S|$.*

An important property of simplices and simplicial cones is the following (This result and the next one are well known, but the proofs are our own).

Proposition 1.7.5 (Unique representation in simplicial cones)**.** *A point x in a simplicial cone C has a unique representation as a nonnegative combination of the generators of K.*

Proof. We will use simplicial cones more, so we prove the result for these. The proof for simplices is essentially the same.

Assume the generators are $\mathbf{v}_1, \ldots, \mathbf{v}_k$ and assume $\mathbf{x} \in C$. Then $\mathbf{x}$ can be written as a nonnegative combination of the generators with weights λ_i. Assume we can write it as another nonnegative combination with weights μ_i, and such that $\lambda_j \neq \mu_j$ for at least one j. Then clearly

$$\mathbf{0} = \mathbf{x} - \mathbf{x} = \sum_{j=1}^{k} \lambda_j \mathbf{v}_j - \sum_{j=1}^{k} \mu_j \mathbf{v}_j = \sum_{j=1}^{k} (\lambda_j - \mu_j)\mathbf{v}_j.$$

Since there is at least one j such that $\lambda_j - \mu_j \neq 0$ this is a contradiction of the assumption that the generators are linearly independent, and so we cannot have two different nonnegative combinations for one particular point $\mathbf{x}$. □

We can now show why simplices and simplicial cones are useful – their facets are easy to find.

Theorem 1.7.6 (Facets of simplicial cones)**.** *Any facet F of a simplicial cone $C = \operatorname{cone} S$ is of the form $F = \operatorname{cone}(S \backslash \{\mathbf{y}\})$ where $\mathbf{y} \in S$.*

Proof. Assume F is of the form $F = \text{cone}(S\backslash\{\mathbf{y}\})$. Clearly $F \subset C$, and F itself is a simplicial cone with $\dim F = \dim K - 1$. If $\mathbf{x}_1, \mathbf{x}_2 \in C$ are such that for some $\lambda \in (0,1)$ the point $\mathbf{z} = (1-\lambda)\mathbf{x}_1 + \lambda\mathbf{x}_2 \in F$ then $\mathbf{z}$ has a unique representation as a nonnegative combination of elements in $S\backslash\{\mathbf{y}\}$ by Proposition 1.7.5. $\mathbf{x}_1$ and $\mathbf{x}_2$ are in C, and as such they both have unique representations as nonnegative combinations of elements in S. Let μ_1 be the coefficient of $\mathbf{y}$ in the representation of $\mathbf{x}_1$ and let μ_2 be the coefficient of $\mathbf{y}$ in the representation of $\mathbf{x}_2$. Then we see that $\mathbf{z}$ can be written as a combination of elements in S with the coefficient of $\mathbf{y}$ equal to

$$(1-\lambda)\mu_1 + \lambda\mu_2.$$

Both μ_1 and μ_2 are nonnegative, so the only way this can be zero is if they too are both zero. If the coefficient is not zero, then $\mathbf{z}$ is not in F by linear independence of S. From this it follows that both $\mathbf{x}_1$ and $\mathbf{x}_2$ are in F, and so F is a face, and by our earlier observation of its dimension, F is a facet.

Now assume F is a facet of C. All elements in C are nonnegative combinations of elements in S, so all elements in F must be so too. It follows from this that F must contain at least some of the generators in S, as F is a facet. Assume I is the index set corresponding to all the generators contained in F. F thus contains $\text{cone}(\{v_i : i \in I\})$. If $F = \text{cone}(\{v_i : i \in I\})$ then $\dim F = |I|$ so by assumption $|I| = |S| - 1$, which means F is of the postulated form.

Now assume instead that F is larger than $\text{cone}(\{v_i : i \in I\})$. Then there is some $\mathbf{x} \in F$ which is not a nonnegative combination of the generators with indices in I. This means $\mathbf{x}$ is a nonnegative combination of some larger set of generators, which by the assumption of F being a facet means that they too should be in F. This contradicts our definition of I above, and so F cannot be larger than the cone. □

The significance of this last result is that any facet of a simplex contains exactly $k-1$ generators, and they also describe the facet completely. What this means, again, is that any facet-defining inequality holds with equality in precisely $k-1$ generators, and the other way around, a valid inequality that holds with equality in precisely $k-1$ generators must be facet-defining (Recall that the set of points for which a valid inequality holds with equality is a face). In Chapter 3 we will use this theorem to find a generating set for the cone of nonnegative diagonally dominant matrices.

1.8 Some graph theory

In the study of completely positive matrices it will be useful to apply some graph theory. We restate the notions we will use here for convenience.

Definition 1.8.1 (Graphs and subgraphs)**.** *A* graph *is a tuple $G = (V, E, I)$, where the set V is called the* vertex set *of the graph (and its elements are called* vertices*), the set E is called the* edge set *of the graph (and its elements are called edges), and I is a function taking any edge to some set consisting of two vertices (or possibly just one). If $I(e) = \{u, v\}$ for $e \in E, u, v \in V$, we say that e is the* edge from u to v*, or the other way around from v to u, as a set has no ordering. Thus our graphs are undirected. We assume that I is injective, so no*

two edges are the same. An edge consisting of only one vertex, say u, will be thought about as an edge from u to u, and referred to as a loop.

In an abuse of notation we will often not bother talking about the function I at all, and instead refer to an edge as a tuple (u, v). It will implicitly be assumed that $(u, v) = (v, u)$, so no confusion should arise.

A subgraph G_0 of G is a tuple $G_0 = (V_0, E_0, I|_{E_0})$. Here $V_0 \subseteq V$, $E_0 \subseteq E$ and $I|_{E_0}$ denotes the restriction of I to E_0. We also require that every edge $e \in E_0$ is between two vertices in V_0, so the subgraph actually is a graph.

Let v be a vertex in V for some graph G. *Removal* of a vertex is a process for obtaining a subgraph. If we remove v we get the subgraph whose vertex set is $V \setminus \{v\}$, and whose edge set consists of all edges in E except those with v as one of their endpoints.

In a graph G a *path* is a sequence whose first and last elements are vertices, every other element is a vertex, and the elements in between are edges between the two adjacent vertices. In a graph where the vertices are represented by letters a path could for instance be $a, (a, b), b, (b, d), d$. This is referred to as a *path from a to d*. A path that is not valid is for instance $a, (x, d), c$, as the edge is not from one of the vertices to the other.

A *connected* graph is a graph such that from every vertex there is at least one path to every other vertex (we consider a graph consisting of one vertex, as well as the empty graph, to be connected). A vertex such that its removal would cause a connected graph to no longer be connected is known as a *cut vertex*. A *block* of a graph is a subgraph which contains no cut vertices and is not properly contained in any other subgraph containing no cut vertices.

A cycle is a connected graph in which every vertex is incident to precisely two edges. For any given positive integer n there is only one cycle (up to relabeling and reordering of vertices) that contains n vertices, and it is referred to as C_n, where n is called the *order* of the cycle. A cycle of odd order is called an *odd cycle*, while a cycle of even order is called an *even cycle*. A *complete* graph is a graph with every possible edge in its edge set, except for loops. A complete subgraph is sometimes called a *clique*.

For a general symmetric $n \times n$ matrix A we define the *graph of A* by letting $V = \{1, 2, \ldots, n\}$ and letting $\{i, j\} \in E$ if and only if $i \neq j, A_{ij} \neq 0$. In other words, the graph is entirely dependent on the zero patterns in the non-diagonal entries. Since the matrices are symmetric no ambiguity arises from the definition. Conversely, a *matrix realisation of a graph G* is a matrix A such that $G(A) = G$.

1.9 Linear programming and mathematical optimisation

We will use linear programming (LP) later in this thesis. It is hopeless to provide anything resembling a short introduction to such a vast subject, so we satisfy ourselves with stating the most fundamental of notions, see for example [16] for

more. A *linear program* is an optimisation problem of the following form:

$$\begin{aligned} &\text{minimise } \mathbf{c}^T\mathbf{x} \\ &\text{subject to } A\mathbf{x} \leq \mathbf{b} \\ &\mathbf{x} \geq \mathbf{0} \end{aligned} \tag{1.3}$$

where we assume $\mathbf{x} \in \mathbb{R}^m$, and $\mathbf{c} \in \mathbb{R}^m, \mathbf{b} \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n \times m}$ are given.

There are several available algorithms for solving linear programs with varying time signatures (See Section 1.10), and linear programs are essentially solvable "efficiently", which is to say in reasonable time compared to the size of the problem. Notable algorithms are the simplex algorithm (of which there are several variants) and the interior-point methods (of which, again, there are several).

For more general minimisation (or maximisation) problems, the situation is more difficult. However, some methods have shown themselves to be valuable, and we will use a so-called steepest descent method in Chapter 5.

1.10 Big O-notation

In this thesis we will discuss some algorithms for determining whether or not a given matrix is completely positive. We will make some mention of an algorithm's efficiency, which is in general a complicated subject. Here we will, following [17], define what so-called big-O notation is, so there will be no confusion about what we mean when we say that an algorithm is "good" or "bad".

Assume we have some function f whose domain is the natural numbers. Then we say that $f(n) = O(g(n))$ for some $g(n)$ if there exist positive constants c and n_0 such that $f(n) \leq cg(n)$ for all $n \geq n_0$. The proper way to think of the definition is that the function $f(n)$ will eventually be dominated by a function of the form $g(n)$, possibly after scaling. If $f(n)$ is some polynomial of degree k, for instance, the x^k term will eventually dominate it, and so in those cases we say that the polynomial is $O(n^k)$.

We do not introduce this notation because we intend to perform detailed studies of runtime characteristics in this algorithm, but in general we wish to make it clear what we mean when we say that one algorithm is better than another. When we say an algorithm is $O(g(n))$ we mean that the time it takes in the worst case is bounded by this expression, where n is the size of our input (for example, the number of variables in an optimisation problem). When we say that one algorithm is better than another, we mean that it runs faster than the other on comparable inputs, which means that one algorithm is bounded in the big-O sense by a function that takes smaller values than is the other.

An algorithm which is $O(g(n))$ with $g(n)$ a polynomial is known as a polynomial-time algorithm. In general, no polynomial-time algorithm is known for determining complete positivity, and the question of whether such an algorithm exists is to the present author's knowledge unsolved.

Chapter 2

Completely Positive Matrices

With preliminary definitions and results out of the way, we can move on to the first proper part of this master's thesis. In this chapter we will define completely positive matrices and attempt to provide an overview of some of the central results in the field. These are all known, and many are taken from [3], but the organisation is largely our own (we heavily emphasise the cone properties of the set of completely positive matrices), and we have provided our own examples as well as clarifying some of the proofs.

Throughout we will try to recall that our primary purpose here is to detail what is known about *when* a matrix is completely positive. The cone of completely positive matrices does have some intrinsic interest too, as we discussed in Section 1.1, but that is, as we said, not the topic of this thesis. Through the entire chapter we will assume n is some fixed positive natural number.

2.1 Completely positive matrices

The results in this section, except where otherwise noted, are found in [3]. We begin by defining what we mean when we say that a matrix is completely positive.

Definition 2.1.1 (Complete Positivity)**.** *If $A \in \mathcal{S}_+^n$ can be decomposed as*

$$A = BB^T$$

where $B \in \mathbb{R}_+^{n\times k}$ for some k, we say that A is completely positive*. We will call such a decomposition a* completely positive decomposition*. The set of completely positive $n \times n$-matrices is denoted by $\mathcal{CP}_n$.*

An example of a completely positive matrix along with its factorisation is

$$\begin{bmatrix} 2 & 5 \\ 5 & 38 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 3 & 2 & 5 \end{bmatrix} \begin{bmatrix} 1 & 3 \\ 1 & 2 \\ 0 & 5 \end{bmatrix}.$$

There are two other easily obtained characterisations of complete positivity. Consider the following, where $\mathbf{b}_i$ is the i-th column of B:

$$A = BB^T = \begin{bmatrix}\mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_k\end{bmatrix} \begin{bmatrix}\mathbf{b}_1^T \\ \vdots \\ \mathbf{b}_k^T\end{bmatrix} = \sum_{i=1}^{k} \mathbf{b}_i \mathbf{b}_i^T. \tag{2.1}$$

It should be obvious that any matrix that has a completely positive decomposition $A = BB^T$ can also be written as $A = \sum_{i=1}^{k} \mathbf{b}_i \mathbf{b}_i^T$ with $\mathbf{b}_i \geq 0$ for $i = 1, \ldots, k$ by defining the $\mathbf{b}_i$ as in (2.1). This latter way of writing A is known as a *rank 1 representation*[1].

We could also partition B into rows, obtaining yet an equivalent definition. Assume A has a completely positive decomposition $A = BB^T$. Then observe the following, where $\widetilde{\mathbf{b}}_i^T$ is the i-th row of B:

$$A = BB^T = \begin{bmatrix}\widetilde{\mathbf{b}}_1^T \\ \vdots \\ \widetilde{\mathbf{b}}_n^T\end{bmatrix} \begin{bmatrix}\widetilde{\mathbf{b}}_1 & \cdots & \widetilde{\mathbf{b}}_n\end{bmatrix} = \begin{bmatrix}\langle \widetilde{\mathbf{b}}_1, \widetilde{\mathbf{b}}_1 \rangle & \cdots & \langle \widetilde{\mathbf{b}}_1, \widetilde{\mathbf{b}}_n \rangle \\ \vdots & \ddots & \vdots \\ \langle \widetilde{\mathbf{b}}_n, \widetilde{\mathbf{b}}_1 \rangle & \cdots & \langle \widetilde{\mathbf{b}}_n, \widetilde{\mathbf{b}}_n \rangle\end{bmatrix}. \tag{2.2}$$

This last matrix is expressed more succinctly as $\mathrm{Gram}(\widetilde{\mathbf{b}}_1, \ldots, \widetilde{\mathbf{b}}_n)$. In this way we see that an equivalent condition for complete positivity is that A is the Gram matrix of nonnegative vectors, specifically vectors in $\mathbb{R}_+^k$. We recall from Theorem 1.5.1 that any positive semidefinite $A \in \mathcal{S}^n$ can be written as the Gram matrix of n vectors in $\mathbb{R}^n$, but these may or may not be nonnegative.

We summarise these equivalent conditions in the following proposition:

Proposition 2.1.2 (Complete Positivity redux)**.** *The following three are equivalent for some $n \times n$ matrix A:*

1. *$A = BB^T$ for some $B \in \mathbb{R}^{n \times k}$.*
2. *$A = \sum_{i=1}^{k} \mathbf{b}_i \mathbf{b}_i^T$, where $\mathbf{b}_i \geq 0$ for $i = 1, \ldots, k$.*
3. *$A = \mathrm{Gram}(\widetilde{\mathbf{b}}_1, \ldots, \widetilde{\mathbf{b}}_n)$ with $\widetilde{\mathbf{b}}_i \in \mathbb{R}_+^k$ for $i = 1, \ldots, n$.*

From a computational viewpoint, note that none of the three conditions seem to be particularly much more useful than the others. The third one, however, offers a new perspective on complete positivity. We follow this train of thought for a short while. If $A = \mathrm{Gram}(\mathbf{v}_1, \ldots, \mathbf{v}_n)$ with $\mathbf{v}_i \in \mathbb{R}^n$, complete positivity of A is really equivalent to asking whether there exists some k and an isometry $T : \mathbb{R}^n \to \mathbb{R}^k$ such that $T\mathbf{v}_i \in \mathbb{R}_+^k$ for all i. Put more informally, A is completely positive if the vectors it is a Gram matrix of can be rotated into the nonnegative orthant of some space which is possibly of higher dimension. Since this is a somewhat important result, we take a moment to prove it. The following proposition is found in [18], as is the proof, which we have clarified quite a bit.

[1] Technically a rank 1 representation as commonly used does not require that the factors be nonnegative, but we will only be using rank 1 representations of completely positive matrices, so we implicitly assume they are nonnegative.

Proposition 2.1.3. *If $A \in \mathcal{S}^n$ is positive semidefinite and has two factorisations*

$$A = BB^T = CC^T \tag{2.3}$$

where $B, C \in \mathbb{R}^{n\times k}$ for some k, there exists an orthogonal matrix $U \in \mathbb{R}^{k\times k}$ such that $C^T = UB^T$.

Proof. Let $\mathbf{b}_i$ be the i-th column of B^T, and let $\mathbf{c}_i$ be the i-th column of C^T. Now let $I \subseteq \{1, \ldots, n\}$ be such that the set $\{\mathbf{b}_i : i \in I\}$ is a basis for the space $\mathrm{Span}(\mathbf{b}_1, \ldots, \mathbf{b}_n)$. Define a linear transformation $S : \mathrm{Span}(\mathbf{b}_1, \ldots, \mathbf{b}_n) \to \mathbb{R}^k$ by $S\mathbf{b}_i = \mathbf{c}_i$ for all $i \in I$. This will then, trivially, have the property that $S\mathbf{b}_i = \mathbf{c}_i$ for $i = 1, \ldots, n$. From 2.3 we know that for all i, j we have

$$a_{ij} = \langle \mathbf{b}_i, \mathbf{b}_j \rangle = \langle \mathbf{c}_i, \mathbf{c}_j \rangle = \langle S\mathbf{b}_i, S\mathbf{b}_j \rangle.$$

Accordingly, S preserves lengths and angles, so S is a linear isometry. We can expand the domain of S outside $\mathrm{Span}(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ without losing this property, so the expanded function T is a linear isometry on $\mathbb{R}^k$. Linear isometries on finite-dimensional spaces can be represented by orthogonal matrices, and the result follows.

We note that it works the other way too: If $C^T = UB^T$ with U orthogonal, $CC^T = BU^TUB^T = BB^T = A$. □

Corollary 2.1.4. *If $A \in \mathcal{DNN}_n$, it can be represented as*

$$A = \mathrm{Gram}(\mathbf{v}_1, \ldots, \mathbf{v}_n),$$

where $\mathbf{v}_i \in \mathbb{R}^n$ for $i = 1, \ldots, n$. Then $A \in \mathcal{CP}_n$ if and only if there exists some k and a linear isometry $T : \mathbb{R}^n \to \mathbb{R}^k$ such that for $T\mathbf{v}_i \in \mathbb{R}^k_+$ for $i = 1, \ldots, n$.

We can use this corollary to turn the question of complete positivity on its head, so to speak. Namely, given a set of vectors in some Euclidean vector space (here $\mathbb{R}^n$) which is such that their inner products are nonnegative, does there exist some other Euclidean vector space (here $\mathbb{R}^k$) such that the vectors can be isometrically embedded in the nonnegative orthant of the latter? The answer is yes if and only if the Gram matrix of the vectors is completely positive. Formulating the question in this way we obtain one possible motivation for studying complete positivity of a matrix beyond the ones we mentioned in Section 1.1.

We have established what positive matrices are, as well as a few different formulations of the condition. We now try to place the set $\mathcal{CP}_n$ in the "matrix universe", as it were. Keep in mind from Section 1.5 that a matrix A having a decomposition $A = BB^T$ where $B \in \mathbb{R}^{n\times k}$ is equivalent to A being positive semidefinite. Definition 2.1.1 only adds the stipulation $B \geq 0$, so all completely positive matrices are positive semidefinite. Further, by construction they are nonnegative, so all completely positive matrices are doubly nonnegative. Unfortunately, doubly nonnegative matrices that are not completely positive do exist – we will return to this later, for now the reader is asked to take it on faith. We have the following relation, where the inclusion is strict.

Proposition 2.1.5. *For any natural number n, $\mathcal{CP}_n \subset \mathcal{DNN}_n$.*

A consequence of Proposition 2.1.5 is that some properties of positive semidefinite matrices carry over. The following is then a corollary of Proposition 2.1.5 and Theorem 1.5.2:

Corollary 2.1.6. *If A is completely positive and of order n, the following holds.*

1. $a_{ij} \leq \frac{1}{2}(a_{ii} + a_{jj})$ *for* $i = 1, \ldots, n-1, j = i+1, \ldots, n$.
2. $a_{ij} \leq \sqrt{a_{ii}a_{jj}}$ *for* $i = 1, \ldots, n-1, j = i+1, \ldots, n$.
3. $\max_{i,j} a_{ij} = \max_i a_{ii}$.
4. $a_{ii} = 0 \Rightarrow a_{ij} = a_{ji} = 0$ *for any fixed* i *with* $j = 1, \ldots, n$.

We recall that $\mathcal{DNN}_n$ is a convex cone, and we might ask ourselves if the set $\mathcal{CP}_n$ has the same property. First, observe the following.

Lemma 2.1.7. *The sum of completely positive matrices is completely positive.*

Proof. Using the rank 1 representations of the matrices involved, this is trivial. □

Lemma 2.1.8. *If $A \in \mathcal{CP}_n$ and $\alpha \in \mathbb{R}^n_+$ then $\alpha A \in \mathcal{CP}_n$.*

Proof. Since $A = BB^T$ where $B \geq 0$, $\alpha A = (\beta B)(\beta B)^T$ where $\beta = \sqrt{\alpha}$ is nonnegative because α is nonnegative, and so βB is a nonnegative matrix, thus proving complete positivity of αA. □

Now we can prove the result we promised earlier, which is our first result of real significance.

Theorem 2.1.9. *$\mathcal{CP}_n$ is a closed convex cone.*

Proof. The convex cone part is easy – it is a direct consequence of Lemma 2.1.7 and Lemma 2.1.8. What remains is to prove closedness.

Assume $\{A_t\}_{t=1}^{\infty}$ is a sequence in $\mathcal{CP}_n$ that converges to some matrix $A = [a_{ij}]_{i,j=1}^n$. We will show that A must also be in $\mathcal{CP}_n$. Being completely positive, each A_i must be a Gram matrix of a set of vectors $\widetilde{\mathbf{b}}_{t,1}, \ldots, \widetilde{\mathbf{b}}_{t,n}$. We consider what happens on the diagonal. Since the sequence converges to A, clearly

$$\lim_{t\to\infty} \langle \widetilde{\mathbf{b}}_{t,i}, \widetilde{\mathbf{b}}_{t,i} \rangle = \lim_{t\to\infty} \|\widetilde{\mathbf{b}}_{t,i}\|^2 = a_{ii}.$$

We start by considering $i = 1$. A convergent sequence is bounded, and since $\{\|\widetilde{\mathbf{b}}_{t,1}\|^2\}_{t=1}^{\infty}$ is bounded, the sequence $\{\|\widetilde{\mathbf{b}}_{t,1}\|\}_{t=1}^{\infty}$ is bounded, which means $\{\widetilde{\mathbf{b}}_{t,1}\}_{t=1}^{\infty}$ is bounded by definition. Any bounded sequence has a convergent subsequence, so we can find some vector $\widetilde{\mathbf{b}}_1$ and a subsequence indexed by the indices t_s with

$$\lim_{s\to\infty} \widetilde{\mathbf{b}}_{t_s,1} = \widetilde{\mathbf{b}}_1.$$

By construction, it follows that $\langle \widetilde{\mathbf{b}}_1, \widetilde{\mathbf{b}}_1 \rangle = a_{11}$. We can proceed by considering the sequence $\widetilde{\mathbf{b}}_{t_s,2}$, and it should be clear to the reader that using the exact same procedure again we can obtain a new reindexing t_{s_r} such that $\widetilde{\mathbf{b}}_{t_{s_r},1}$ converges to $\widetilde{\mathbf{b}}_1$ and $\widetilde{\mathbf{b}}_{t_{s_r},2}$ converges to some limit $\widetilde{\mathbf{b}}_2$. We can proceed in this manner until we have found limits $\widetilde{\mathbf{b}}_1, \ldots, \widetilde{\mathbf{b}}_n$ with the property that, for all i, $a_{ii} = \langle \widetilde{\mathbf{b}}_i, \widetilde{\mathbf{b}}_i \rangle$. It should also be clear that by construction, $a_{ij} = \langle \widetilde{\mathbf{b}}_i, \widetilde{\mathbf{b}}_j \rangle$. Then $A = \text{Gram}(\widetilde{\mathbf{b}}_1, \ldots, \widetilde{\mathbf{b}}_n)$. Furthermore, every $\widetilde{\mathbf{b}}_i$ is a limit of nonnegative vectors, and so is itself nonnegative. Thus, A fulfills condition 3 in Proposition 2.1.2, and is completely positive. □

Cones are often described in terms of their extreme rays, as mentioned in Section 1.6. In, for instance, [10], the following result is given:

Lemma 2.1.10. *The extreme vectors of $\mathcal{CP}_n$ are the matrices in the set*

$$Ext = \{\mathbf{x}\mathbf{x}^T : \mathbf{x} \in \mathbb{R}^n_+\}.$$

The extreme rays are then obtained from this set by identifying matrices that are scalar multiples of each other.

Proof. By condition 2 in Proposition 2.1.2, obviously

$$\mathcal{CP}_n \subseteq \text{conv}(\text{Ext}).$$

Furthermore, if $A \in$ Ext, it can only be written as a convex combination of two other elements of Ext if they are in fact multiples of A. From this we can conclude that the set in the lemma is in fact the set of extreme vectors of $\mathcal{CP}_n$. □

There is one important cone which is contained within the cone of completely positive matrices. Recall from Chapter 1 that a diagonally dominant matrix is a matrix $A = [a_{ij}]_{i,j=1}^n$ such that

$$a_{ii} \geq \sum_{k=1,k\neq i}^{n} a_{ij} \text{ for } i = 1, \ldots, n.$$

holds. Note that if we wish to require nonnegativity as well, the only inequalities we have to add are

$$a_{ij} \geq 0 \text{ for } i = 1, \ldots, n-1, j = i+1, \ldots, n,$$

under the assumption that A is symmetric.

Proposition 2.1.11. *The set of all matrices in $\mathcal{S}^n_+$ that are diagonally dominant, denoted hereafter by $\mathcal{DD}_n$, is a convex cone.*

Proof. This is, essentially, obvious. Note that diagonal dominance and nonnegativity is obviously retained when scaling by a nonnegative real number. Furthermore, diagonal dominance is defined in terms of linear inequalities, so any nonnegative sum of matrices satisfying these inequalities will also satisfy them, thus proving the cone property. □

We can now prove the following theorem.

Theorem 2.1.12. *For all n, $\mathcal{DD}_n \in \mathcal{CP}_n$.*

Proof. Assume $A = [a_{i,j}]_{i,j=1}^n \in \mathcal{DD}_n$. Let D_i be the matrix $\mathbf{e}_i\mathbf{e}_i^T$ for $i = 1, \ldots, n$ and let E_{ij} be $(\mathbf{e}_i + \mathbf{e}_j)(\mathbf{e}_i + \mathbf{e}_j)^T$ for $i = 1, \ldots, n-1, j > i$. Finally, let

$$r_i = a_{ii} - \sum_{j\neq i} a_{ij}.$$

This number is nonnegative by the condition of diagonal dominance. All the matrices D_i and E_{ij} are completely positive by construction, and it should also be readily apparent that

$$A = \sum_{i=1}^{n} r_i D_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} a_{ij} E_{ij},$$

so A is a completely positive matrix, being a sum of completely positive matrices with nonnegative weights. □

Could it be that every completely positive matrix is diagonally dominant? No, consider the following matrix, which is in $\mathcal{CP}_3$, but not in $\mathcal{DD}_3$:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}. \tag{2.4}$$

This shows that there are completely positive matrices that are not in the cone of diagonally dominant matrices. Thus, our chain of reasoning shows that for any positive integer n,

$$\mathcal{DD}_n \subset \mathcal{CP}_n \subset \mathcal{DNN}_n \subset \mathcal{S}_+^n. \tag{2.5}$$

All the above inclusions are strict. We have not explicitly mentioned the last one, but it is obvious provided we simply break one of the necessary conditions in Theorem 1.5.2. As an example consider the following matrix:

$$\begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}.$$

Simple inspection shows that this is not a positive semidefinite matrix, but it is certainly symmetric and nonnegative, thus proving our claim.

A few simple facts about $\mathcal{CP}_n$ remain to be proved. First, we can obtain some if-and-only-if conditions for complete positivity by considering certain symmetric matrix products.

Lemma 2.1.13. *If A is an $n \times n$ completely positive matrix, and C is an $m \times n$ nonnegative matrix, then CAC^T is completely positive.*

Proof. Assume $A = BB^T, B \geq 0$. then $CAC^T = (CB)(CB)^T$. □

Lemma 2.1.14. *Let A be an $n \times n$ matrix. Then the following hold:*

1. *If P is an $n \times n$ permutation matrix, then A is completely positive if and only if PAP^T is completely positive.*

2. *If D is a positive diagonal $n \times n$ matrix, then A is completely positive if and only if DAD is completely positive.*

Proof. Note that in both cases, one implication follows immediately from Lemma 2.1.13. We prove the other implications.

1: Assume $PAP^T = BB^T$. Since P is a permutation matrix, $PP^T = P^TP = I$, so we multiply from the left by P^T and from the right by P. Then we get that

$A = P^T BB^T P = (P^T B)(P^T B)^T$, which is a completely positive decomposition of A.

2: Assume $DAD = BB^T$. Assume the diagonal elements of D are $d_1, \ldots, d_n$. Since D is a positive diagonal matrix, these are all positive, and so

$$\text{diag}(1/d_1, \ldots, 1/d_n) = D^{-1}$$

exists and is positive diagonal. Now we multiply by D^{-1} from both sides, obtaining $A = D^{-1}BB^T D^{-1} = (D^{-1}B)(D^{-1}B)^T$, which is a completely positive decomposition of A. □

In the lead-up to Theorem 2.1.9 we showed that the set $\mathcal{CP}_n$ is closed under addition and multiplication by nonnegative scalars. In fact we can expand slightly upon this.

Lemma 2.1.15. *The m-th power of an $n \times n$ completely positive matrix A is completely positive for any m.*

Proof. If m is even, $m = 2t$ for some t, so $A^m = (A^t)^2 = (A^t)(A^t)^T$, which is a completely positive decomposition. If $m = 2t + 1$, $A^m = (A^t)A(A^t)^T$, and the result follows from lemma 2.1.13. □

For any polynomial $f(x) = a_m x^m + a_{m-1}x^{m-1} + \ldots + a_1 x + a_0$ with domain $\mathbb{R}$, we can define a polynomial with domain $\mathbb{R}^{n\times n}$ by letting $f(X) = a_m X^m + a_{m-1}x^{m-1} + \ldots + a_1 X + a_0 I_n$, where all of the matrix powers are defined because the matrices are square. We can now state the following corollary.

Corollary 2.1.16. *If $f(x)$ is a real polynomial with nonnegative coefficients and A is completely positive, then $f(A)$ is also completely positive.*

Proof. This is a direct consequence of Lemma 2.1.8, Lemma 2.1.7 and Lemma 2.1.15. □

This concludes our preliminary investigation of complete positivity.

2.2 Further results

Earlier we said that $\mathcal{CP}_n$ is strictly contained in $\mathcal{DNN}_n$. If one tries to prove this, say, by finding a small matrix that is doubly nonnegative yet not completely positive, one quickly runs into trouble, as none readily appear. This is seen in the following theorem. As in the previous section, we draw from [3] throughout except where otherwise mentioned. We have clarified some of the proofs.

Theorem 2.2.1. *For $n \leq 4$, $\mathcal{DNN}_n = \mathcal{CP}_n$.*

We will prove this by proving the cases $n = 1, n = 2, n = 3$ separately (we do not have room for the proof of the case $n = 4$, it can be found in [3] for the interested reader).

Lemma 2.2.2. *If $A \in \mathcal{DNN}_n$ has rank 1, $A \in \mathcal{CP}_n$.*

Proof. If A has rank 1 it can be written as $\mathbf{b}\mathbf{b}^T$ for some column vector $\mathbf{b} \in \mathbb{R}^n$. Then, since A is nonnegative, simple inspection of the elements in $\mathbf{b}\mathbf{b}^T$ shows that all the elements of $\mathbf{b}$ must have the same sign. If they are all nonnegative, we are done. If not, they are all nonpositive, and in that case $A = (-\mathbf{b})(-\mathbf{b}^T)$ is a completely positive decomposition of A. □

Note that Lemma 2.2.2 is quite a bit stronger than we needed to prove that $\mathcal{DNN}_1 = \mathcal{CP}_1$ (which we could really have done merely by taking a square root), but this more general form is interesting – that all rank 1 doubly nonnegative matrices are completely positive is better than one might expect.

Lemma 2.2.3. *If $A \in \mathcal{DNN}_n$ has rank 2, $A \in \mathcal{CP}_n$.*

Proof. Since A has rank 2, it follows from Theorem 1.5.1 that A is the Gram matrix of n vectors in $\mathbb{R}^2$. If the angle between any of the two vectors were greater than $\frac{\pi}{2}$ the inner product between those would be negative, so the maximal angle has to be $\frac{\pi}{2}$. Thus we could pick the two vectors that are farthest from each other, and rotate them into $\mathbb{R}^2_+$. In this rotation all the other vectors also end up in the nonnegative quadrant, and as it is a rotation their inner products are not changed, and we have obtained a representation of A as a Gram matrix of nonnegative matrices, thereby proving that A is completely positive. □

Again, this is quite a bit stronger than we needed to prove that $\mathcal{DNN}_2 = \mathcal{CP}_2$. We could hope that the Gram matrix-based approach from the two preceding two lemmas remained viable for higher ranks, but unfortunately it soon gets more complicated – as we shall see in Section 2.4 we cannot, in general, hope to remain in a space of the same dimension when isometrically embedding the Gram vectors in a positive orthant. The next proof only works for matrices of order 3.

Lemma 2.2.4. *If $A \in \mathcal{DNN}_3$, then $A \in \mathcal{CP}_3$.*

Proof. If A has rank 1 or 2, the result follows from the previous two lemmas. Otherwise we can assume A is the Gram matrix of three vectors in $\mathbb{R}^3$, and that they are all linearly independent, which is to say, they are not in the same plane, and all of them are nonzero.

So let $A = \text{Gram}(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$ and let $S = \text{Span}\{\mathbf{v}_1, \mathbf{v}_2\}$. Let $\mathbf{v}_3'$ be the orthogonal projection of $\mathbf{v}_3$ onto S and let $\mathbf{v}_3'' = \mathbf{v}_3 - \mathbf{v}_3'$. Note that by the definition of orthogonal projections,

$$\mathbf{v}_3' = \frac{\langle \mathbf{v}_3, \mathbf{v}_1 \rangle}{\langle \mathbf{v}_1, \mathbf{v}_1 \rangle} \mathbf{v}_1 + \frac{\langle \mathbf{v}_3, \mathbf{v}_2 \rangle}{\langle \mathbf{v}_2, \mathbf{v}_2 \rangle} \mathbf{v}_2,$$

so the matrix $\text{Gram}(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3')$ is completely positive with rank 2.

We could now use the Gram-Schmidt procedure[2] to obtain an orthonormal basis $E = \{\mathbf{b}_1, \mathbf{b}_2\}$ for S, with the property that the coordinate vectors $[\mathbf{v}_1]_E$, $[\mathbf{v}_2]_E$ and $[\mathbf{v}_3']_E$ are nonnegative, and $\mathbf{b}_1 = \mathbf{v}_1/\|\mathbf{v}_1\|$. Let $\mathbf{b}_3 = \mathbf{v}_3/\|\mathbf{v}_3\|$. Then $E' = \{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\}$ is an orthonormal basis for $\mathbb{R}^3$ with $[\mathbf{v}_i]_{E'} \geq 0$ for $i = 1, 2, 3$, which was what we needed to find. □

[2]This is detailed in most elementary texts on linear algebra, and we assume the reader is familiar with it.

It is possible to construct a similar proof for a 4×4-matrix that is doubly nonnegative, the main idea is showing that it suffices to consider a matrix with at least one zero, and then consider which graphs it can have (see Section 1.8). We will not include the proof here, but it can be found in [3] (in which there is both an algebraic and a geometric version).

So with the little note that we left out a small part of the proof, we consider Theorem 2.2.1 proved. A simple example of a matrix in $\mathcal{CP}_3$ is

$$A = \begin{bmatrix} 8 & 3 & 1 \\ 3 & 5 & 4 \\ 1 & 4 & 5 \end{bmatrix}.$$

This matrix was originally obtained by computing

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

but this is not a completely positive factorisation. Nevertheless, as the matrix is in $\mathcal{DNN}_3$, it must be in $\mathcal{CP}_3$. In this case, we are lucky – a simple Cholesky factorisation shows us that if we let

$$B = \begin{bmatrix} 2.8284 & 0 & 0 \\ 1.0607 & 1.9865 & 0 \\ 0.3536 & 1.8415 & 1.2181 \end{bmatrix},$$

then $A = BB^T$ (disregarding rounding errors). The proofs given in connection with Theorem 2.2.1 do suggest algorithms (in fact the proofs can be written out more directly, in an algebraic fashion, but we felt the geometric approach was more interesting), so for small matrices the question of complete positivity is solved.

Theorem 2.2.1 does imply slightly more than it seems to say. Consider, for example, a matrix where the only nonzero elements are blocks on the diagonal, with all blocks being smaller than 4×4 – this is clearly a completely positive matrix if and only if every block is completely positive, which they are if and only if they are doubly nonnegative. Admittedly this situation is a little contrived, and in general we must look for more robust relations. Note, however, that the "algorithm" we suggested just now depends on the zero pattern of the matrix. As it turns out, there is more to say about the connection between zero patterns and complete positivity, and we will talk about that in Section 2.3.

For now, we will move on to another interesting result, due to [18] (the proof is also taken from there, the result is Theorem 2 in the article). The proof uses the notion of a dual cone. If C is a cone in some Euclidean space V, its dual cone C^* is a cone in the same space, defined by:

$$C^* = \{\mathbf{y} \in V : \langle \mathbf{y}, \mathbf{x} \rangle \geq 0 \text{ for all } \mathbf{x} \in C\}.$$

We can now state and prove our result.

Theorem 2.2.5. *Let $A \in \mathcal{DNN}_n$, then $A = BB^T$ with $B \in \mathbb{R}^{m \times k}$, where $k = \operatorname{rank}(A)$. Let C be the cone generated by the columns of B^T (that is, the rows of B, but we will treat them as column vectors). Then $A \in \mathcal{CP}_n$ if and only if there exist m nonzero vectors $\mathbf{x}_1, \ldots, \mathbf{x}_m \in C^*$ such that*

$$\mathbf{x}_1 \mathbf{x}_1^T + \ldots + \mathbf{x}_m \mathbf{x}_m^T = I. \tag{2.6}$$

Proof. First, suppose there exist vectors such that 2.6 holds. Then

$$A = BB^T = BIB^T = B(\mathbf{x}_1\mathbf{x}_1^T + \ldots + \mathbf{x}_m\mathbf{x}_m^T)B^T.$$

Now, this is easily seen to be equal to

$$A = (B\mathbf{x}_1)(B\mathbf{x}_1)^T + \ldots + (B\mathbf{x}_m)(B\mathbf{x}_m)^T,$$

and since each $\mathbf{x}_i$ is in the dual cone generated by the rows of B, it follows that $B\mathbf{x}_i \geq 0$ for $i = 1, \ldots, m$, and therefore this is a completely positive representation of A, thereby proving sufficiency.

Now we wish to prove that the condition is necessary, assume $A \in \mathcal{CP}_n$. Then there exists some nonnegative $n \times m$-matrix E for some m such that $A = EE^T$. By assumption, A has another factorisation as $A = BB^T$ where the number of columns in B is equal to the rank of A – let us call it k. Now, in general E will have more columns than B, but not less (See Section 2.4), so let us modify B by putting $B_1 = \begin{bmatrix} B & O \end{bmatrix}$ where the zero matrix has dimensions $n \times (m-k)$ such that B_1 and E have the same number of columns. Now,

$$A = EE^T = B_1B_1^T, E \in \mathbb{R}_+^{n\times m}, B_1 \in \mathbb{R}^{n\times m},$$

and using Proposition 2.1.3, there exists some orthogonal matrix U such that $E^T = UB_1^T$. We write U as a block matrix so that the parts of U match up with B, that is:

$$UB_1^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} B^T \\ O \end{bmatrix},$$

where $U_1 \in \mathbb{R}^{m\times k}, U_2 \in \mathbb{R}^{m\times(m-k)}$. Then it follows that $E^T = U_1B^T$. Since $U^TU = I$ by definition, $U_1^TU_1 = I_m$. Let $\mathbf{x}_i$ be the i-th column of U_1^T (alternately, the i-th row of U_1 as a column vector). Then this means that

$$\mathbf{x}_1\mathbf{x}_1^T + \ldots \mathbf{x}_m\mathbf{x}_m^T = I_m.$$

All we need to prove is that these vectors are in the dual of the cone generated by the rows of B. Since $E^T = U_1B^T$, $E = BU_1^T$, and since E is nonnegative each $\mathbf{x}_i$ has a nonnegative inner product with every row of B, thus proving this last statement. □

In the entire thesis, this is the only alternate characterisation of completely positive matrices that we will present – all other results are either sufficient or necessary, but not both. Unfortunately, as we see, the equivalent characterisation does not on the surface of it appear simpler to use in practice: How do we determine whether the identity matrix is contained in the dual cone of the columns of B^T? Nevertheless, it is certainly an interesting theorem.

Finally, we will prove one more condition that has to do with complete positivity – this time it is a sufficient one. The theorem as well as its proof are in [3]. Recall from Chapter 1 that the comparison matrix $M(A)$ of A is the matrix defined by

$$M(A)_{ij} = \begin{cases} |A_{ij}| & \text{if } i = j, \\ -|A_{ij}| & \text{otherwise.} \end{cases}$$

Theorem 2.2.6. *If A is symmetric and nonnegative and $M(A)$ is positive semidefinite, A is completely positive.*

Proof. If $M(A)$ is positive semidefinite, there exists a positive diagonal matrix D such that $DM(A)D$ is diagonally dominant – we mentioned this fact without proof in Section 1.3. The absolute values in the elements of $M(A)$ are the same as the ones in A, and by this DAD is diagonally dominant, and by assumption it is symmetric and nonnegative. Hence DAD is completely positive by Theorem 2.1.12, and by Lemma 2.1.14 it follows that A is completely positive. □

In the next section we shall see that there is a certain situation in which the condition of Theorem 2.2.6 is necessary as well as sufficient. It depends on the zero pattern of the matrix.

2.3 Complete positivity and graph theory

In this section we investigate the connection between a matrix, its graph, and complete positivity. As with so many of the other things we talk about, this has been covered in [3], and the results in this section are from there, as well as the proofs.

The best way to see that graph theory is useful to us is by example. By a *triangle-free* graph we mean a graph that contains no triangles, that is, cycles of order 3.

Theorem 2.3.1. *If A is completely positive and $G(A)$ is triangle-free, $M(A)$ is positive semidefinite.*

Proof. Consider a rank 1 representation of A, which we can get by complete positivity:

$$A = \sum_{i=1}^{k} \mathbf{b}_i \mathbf{b}_i^T.$$

Obviously any $\mathbf{b}_i$ will give rise to a clique in the graph of A, and since any clique of size three or larger will contain a triangle none of these can be larger than two. This means that none of the $\mathbf{b}_i$ can have a support larger than 2. Now, for every i such that $|\operatorname{supp} \mathbf{b}_i| = 1$ let $\mathbf{d}_i = \mathbf{b}_i$, while for every i such that $|\operatorname{supp} \mathbf{b}_i| = 2$ let $\mathbf{d}_i$ be obtained from $\mathbf{b}_i$ by changing the sign of one of the two elements in the support. Then it should be obvious that

$$M(A) = \sum_{i=1}^{k} \mathbf{d}_i \mathbf{d}_i^T,$$

and this is enough to prove that $M(A)$ is positive semidefinite according to Theorem 1.5.1. □

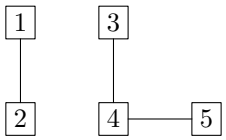
The significance of Theorem 2.3.1 is in relation to Theorem 2.2.6. Put together they give us the following corollary:

Corollary 2.3.2. *If A is symmetric and nonnegative and $G(A)$ is triangle-free, A is completely positive if and only if $M(A)$ is positive semidefinite.*

As an example, consider the following matrix, which is nonnegative and symmetric:

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 1 & 0 \\ 0 & 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 2 & 4 \end{bmatrix}. \tag{2.7}$$

The graph of this matrix is



This is a triangle-free graph, so checking $M(A)$ for positive semidefiniteness will suffice to decide whether or not A is completely positive. However,

$$\begin{bmatrix} 0 & 0 & 1 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 2 & 0 & 0 & 0 \\ 0 & 0 & -3 & 1 & 0 \\ 0 & 0 & -1 & 1 & -2 \\ 0 & 0 & 0 & -2 & 4 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 2 \\ 1 \end{bmatrix} = -1, \tag{2.8}$$

which means that $M(A)$ is not positive semidefinite, and so A is not completely positive. Keep in mind that this is one of those rare cases in which we can say conclusively that a matrix is not completely positive.

We make a small note before moving on: The graph of a matrix (as we use the term) is entirely dependent on its zero pattern outside the diagonal. Therefore, Theorem 2.3.1 appears to be a result concerning the zero pattern of the matrix A. A closer look at the proof is warranted, however, for the proof only depends on the zero pattern of the matrix B in the completely positive decomposition $A = BB^T$. Consider the following matrix:

$$A = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} = BB^T. \tag{2.9}$$

This is a completely positive matrix by construction, and its graph is a triangle – so it certainly is not triangle-free. However, the proof of Theorem 2.3.1 would go through for this matrix, as each of the columns in B at most have two nonzero elements. The reason, presumably, why the theorem is not given for this case in [3] is that we are looking for conditions under which a matrix is completely positive, and this slightly more encompassing condition presupposes knowledge of the completely positive decomposition, while the triangle-free condition only requires knowledge of the matrix we are examining.

We have already seen that studying the graph of a matrix can provide us with some information about whether or not the matrix is completely positive. As it turns out, a lot more can be said than what we have mentioned thus far.

Definition 2.3.3. *A graph G is said to be* completely positive *if every doubly nonnegative matrix realisation of G is completely positive.*

In the rest of the chapter we will assume implicitly that all matrix realisations are doubly nonnegative.

The main result as concerns completely positive graphs is the following – by a *long odd cycle* we mean a cycle of odd length greater than 4.

Theorem 2.3.4. *A graph G is completely positive if and only if it does not contain a long odd cycle.*

Proving this result is quite a bit of work – it is shown in detail (and in more than one way) in [3]. Here, we will restate the intermediate results as lemmas, and how to combine them to prove Theorem 2.3.4 will hopefully be clear. In the interest of saving space and maintaining focus on the ideas, we only sketch the thrust of the proof for the various results.

Lemma 2.3.5. *A graph that contains a long odd cycle is not completely positive.*

Sketch of proof. For any odd n greater than 3, define the $n \times (n-1)$-matrix $B = [b_{ij}]_{i,j=1}^{n,n-1}$ by $b_{ii} = 1$ for $i = 1, \ldots, n-1$, $b_{i+1,i} = 1$ for $i = 1, \ldots, n-2$ and finally $b_{ni} = (-1)^{i+1}$ for $i = 1, \ldots, n-2$, and leave all other elements as zero. Then the matrix $A = BB^T$ is positive semidefinite by construction, nonnegative, and its graph is a long odd cycle. Since it does not contain any triangles, it is completely positive if and only if $M(A)$ is positive semidefinite according to Corollary 2.3.2, and inspection shows that this is not the case.

This shows that a graph is not completely positive if it is a long odd cycle, and if we pick an arbitrary graph G containing one, we can pick a matrix realisation of G such that the submatrix corresponding to the long odd cycle is A from before. We can then create a sequence such that every element in the sequence has G as its graph, and which converges to a matrix whose only nonzero elements are the ones in the long odd cycle. Since this latter matrix is not completely positive, and the cone $\mathcal{CP}_n$ is closed by Theorem 2.1.9, there must be some element in the sequence that is in $\mathcal{DNN}_n$, but not in $\mathcal{CP}_n$. Since this element by construction is a matrix realisation of G, G is not a completely positive graph, and we are done. □

This proves one part of our result, namely that graphs containing long odd cycles are not completely positive. The next part is essentially a consideration of what kinds of graphs do not contain long odd cycles, and then a piece-by-piece proof that they are all completely positive. We also need a result that glues them together, which is the following.

Lemma 2.3.6. *If a graph G is such that $G = G_1 \cup G_2$ where G_1 and G_2 only have one vertex in common (which would be a cut vertex) and G_1 and G_2 are completely positive, then G is completely positive.*

Sketch of proof. In this case, when we pick some matrix realisation of G, we know it is a Gram matrix. We can divide the Gram vectors into three sets, two of which are orthogonal sets, and such that the final set is one single vector which corresponds to the one vertex that is common to the two parts of the graph. The rest of the proof is a technical exercise in which a particular space is created in which we can construct a nonnegative set of Gram vectors that give the same matrix as the one we started with, relying mainly on projection and direct sums of spaces. □

Using Lemma 2.3.6 repeatedly the question of complete positivity of any graph becomes a question of complete positivity of its blocks. What kinds are there?

Lemma 2.3.7. *In a matrix G that does not contain a long odd cycle, every block is either bipartite, T_n or the complete graph on 4 vertices.*

Sketch of proof. This is a fairly straightforward proof by contradiction – if the block has more than 4 vertices, it must be either bipartite or T_n. If it is not, two cycles of length greater than 3 can be found such that one is precisely one step longer than the other, and so one of them must be a long odd cycle. □

Now, the three kinds of blocks must be considered.

Lemma 2.3.8. *The following three types of graphs are all completely positive:*

1. *The complete graph on 4 vertices.*
2. *T_n for any n.*
3. *Any bipartite graph.*

Sketch of proof. The first result is a direct corollary of Theorem 2.2.1. We noted there that the final part of the proof (for $n = 4$) depends slightly on graph theory, so if the reader is concerned about cyclical reasoning it may be reassuring to be reminded that there is a proof of the theorem which is entirely algebraic, see [3].

Proving the other two results is fairly immediate as well, the graph structure in both cases makes rather strict demands on their matrix realisations, and it is mostly an algebraic exercise. The proof for T_n depends on the use of Schur complements, while the proof for bipartite graphs depends on proving that the comparison matrix is positive semidefinite. □

Finally, we see that Theorem 2.3.4 follows immediately from Lemmas 2.3.5, 2.3.6, 2.3.7 and 2.3.8, thus concluding our sketch of its proof.

We consider a small example. Let S be the following set of vectors in $\mathbb{R}^6$:

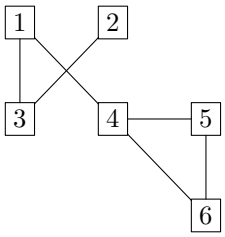
$$S = \left\{ \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ -5 \\ 0 \\ 1 \\ 1 \\ -2 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 2 \\ 3 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 4 \\ 0 \\ -1 \end{bmatrix} \right\}. \tag{2.10}$$

Now define the following matrix.

$$A = \operatorname{Gram} S = \begin{bmatrix} 2 & 0 & 1 & 5 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 2 & 0 & 0 & 0 \\ 5 & 0 & 0 & 31 & 5 & 6 \\ 0 & 0 & 0 & 5 & 13 & 8 \\ 0 & 0 & 0 & 6 & 8 & 17 \end{bmatrix}. \tag{2.11}$$

This matrix is not (to our knowledge) the Gram matrix of nonnegative vectors, $M(A)$ is not positive semidefinite, and while A is doubly nonnegative, it is too

large to apply Theorem 2.2.1. We note that it is relatively sparse, however, and seems to have a certain block diagonal-like property. The graph of A looks like this:



The main object of interest here is that the graph does not contain any long odd cycles. Thus, since A is doubly nonnegative, it follows from Theorem 2.3.4 that A must be a completely positive matrix. This proves that the theorem does give us something. It is not evident from the presentation we have given, but the proofs of Theorem 2.3.4 are in fact constructive, so using the theory we could construct a completely positive decomposition of the matrix A should we so wish.

We note that there is much more to say about the relationship between graphs and completely positive matrices, but we shall let the matter lie here – Theorem 2.3.4 seems to be by far the most striking product of this part of the theory.

2.4 CP-rank

Something we have hinted at before is that in general the matrix B in a completely positive decomposition will have more columns than its relative in a positive semidefinite decomposition. This is not in the core of our thesis, but we state some of the most important results without proof. The results are again taken from [3] except where otherwise noted.

Definition 2.4.1. *Let A be an $n \times n$ completely positive matrix. The minimal k such that $A = BB^T$ for some $B \in \mathbb{R}_+^{n\times k}$ is called the* cp-rank *of A. The cp-rank of A is denoted by* $\text{cp-rank}\, A$.

Our first result is the following important one.

Proposition 2.4.2. *For any completely positive matrix A,*

$$\text{cp-rank}\, A \geq \text{rank}\, A.$$

This is self-evident – if the cp-rank could be less than the rank, it would mean the rank of A was actually smaller than we had assumed, see Theorem 1.5.1.

The next result is also trivial – simply consider the completely positive rank 1 representations of the matrices involved.

Proposition 2.4.3. *If A and B are both in $\mathcal{CP}_n$,*

$$\text{cp-rank}(A + B) \leq \text{cp-rank}\, A + \text{cp-rank}\, B.$$

In Section 2.1 we showed that the property of complete positivity is invariant under certain transformations. The next two results show that cp-rank also plays rather nicely with these.

Proposition 2.4.4. *If S is a nonsingular matrix with a nonnegative inverse and SAS^T is completely positive, A is also completely positive and*

$$\text{cp-rank}\, A \leq \text{cp-rank}(SAS^T) \tag{2.12}$$

We mention the next result because it concerns precisely the type of operations we considered in Lemma 2.1.14.

Proposition 2.4.5. *Let A be a completely positive matrix, D be a positive diagonal matrix and P be a permutation matrix. Then*

$$\text{cp-rank}\, A = \text{cp-rank}\, DAD = \text{cp-rank}\, PAP^T.$$

The final, and most striking, theorem in this section is taken from [1].

Theorem 2.4.6. *If A is completely positive and has rank $r \geq 2$, then*

$$\text{cp-rank}\, A \leq \frac{r(r+1)}{2} - 1.$$

Further, for any such rank r, there exists a completely positive matrix with rank r and cp-rank $\frac{r(r+1)}{2} - 1$, in other words, the bound is tight.

The reader may recall that in proving Theorem 2.2.1 we mentioned that the process of obtaining a completely positive decomposition of a small matrix can be made explicit through an algebraic proof, and when following said proof it turns out that for any small matrix A that is completely positive we can find a square matrix $B \geq 0$ such that $A = BB^T$. Therefore, in general, we cannot always find a matrix of order n with cp-rank equal to $\frac{n(n+1)}{2} - 1$. It has been conjectured that the cp-rank of of matrices of order n is bounded by $\frac{n^2}{4}$, but to the author's knowledge this has not been proved.

2.5 Summary and comments

The theory of completely positive matrices is extensive, perhaps surprisingly so, and it would be impossible to go over it all here. Besides, [3] already exists, which is a very good book covering much of the theory that had been published up until its writing in 2006. Our aim, as mentioned, has only been to cover the most important results in the theory, either because they are interesting in and of themselves or because they may prove useful to us later on.

We summarise the chapter informally in the following little list, hopefully providing a useful overview of the theory we have covered.

- A matrix A is completely positive if it can be written as $A = BB^T$, where $B \geq 0$. Equivalently, A is completely positive if it can be written as a sum of nonnegative symmetric rank 1 matrices, or as the Gram matrix of nonnegative vectors.

- If A has rank r, the minimal number of columns in B is bounded above by $\frac{r(r+1)}{2} - 1$, and is called the cp-rank of A. In a rank 1 representation, this same number bounds the minimal number of rank 1 matrices, while in a Gram matrix representation, it bounds the minimal dimension of the space we find the Gram vectors in. The cp-rank is bounded below by the rank.

- The set of completely positive matrices, denoted $\mathcal{CP}_n$, is a closed convex cone which is properly contained in the set of doubly nonnegative matrices, $\mathcal{DNN}_n$.

- We have shown a couple of ways of deciding whether or not a matrix A in $\mathcal{DNN}_n$ is completely positive:

  1. If A is diagonally dominant, A is completely positive.
  2. If $n \leq 4$, A is completely positive.
  3. If $M(A)$ is positive semidefinite, A is completely positive. If $G(A)$ is triangle-free, this is also sufficient.
  4. If $G(A)$ is a graph with no long odd cycles, A is completely positive.

  Note that all of these tests actually do give us the completely positive representation. Technically, we might include Cholesky factorisation in this list, but in general we can rarely hope that the Cholesky factor is nonnegative, so this is not a very good test.

Some results that are interesting, but we have not mentioned, include [12], in which Theorem 2.4.6 is obtained through the use of linear programming and the simplex algorithm and [4], in which matrices that can be written as BB^T where B is a matrix consisting only of zeroes and ones are studied as a form of complete positivity. Results similar to those for small completely positive matrices are discovered, and there too the problem proves more difficult for matrices of order 5 or greater.

One very interesting result is in [2], in which an algorithm for determining the CP-rank of a matrix is given. This algorithm will return ∞ if the matrix is not completely positive, so this is in fact a proper test for complete positivity. Unfortunately, it requires something in the order of $O((n^2)^{n^3})$ operations, so it is hardly practical. It does, however, encouragingly serve to show that the problem of complete positivity is not entirely unsolvable.

In the next chapter we will see if we can obtain some interesting conclusions from the relation $\mathcal{DD}_n \subset \mathcal{CP}_n$, which we showed in Section 2.1.

Chapter 3

Expansions of the Diagonally Dominant Cone

3.1 Preliminaries

Throughout this chapter we will be using results from Sections 1.6 and 1.7. In those sections we usually stated the results for spaces of vectors, because that is easier to explain, but in this chapter we shall be using the results on spaces of matrices. The generalisation is obvious – $\mathbb{R}^{n\times m}$ and $\mathbb{R}^{nm}$ are topologically the same space.

Theorem 2.1.12 says that $\mathcal{DD}_n \subset \mathcal{CP}_n$. One good thing about the cone $\mathcal{DD}_n$ is that testing for membership is quick – we merely have to verify n linear inequalities, each in n variables, which means the process is $O(n)$. Perhaps it would be possible to find a larger cone than $\mathcal{DD}_n$ which is still contained in $\mathcal{CP}_n$? The test for membership (which will involve verifying some number of inequalities) might even be quick. Searching for such a cone will be the overarching aim in this chapter.

In order to accomplish our goal, we will first obtain the generators of $\mathcal{CP}_n$, then add to them some more completely positive generators, and finally go the other way again, obtaining the defining linear inequalities of the cone defined by this larger set of generators. As long as all the generators are completely positive, the entire cone is, by Lemmas 2.1.7 and 2.1.8. According to Theorems 1.7.1-1.7.3 we can obtain the unique minimal inequality system by looking for facet-defining inequalities ($\mathcal{DD}_n$ is already full-dimensional in the space $\mathcal{S}^n_+$, so when expanding it we will certainly retain full-dimensionality). Note that while any cone we study in this fashion will be completely positive, much rests upon whether or not we can explicitly describe the linear inequality system – otherwise we can hardly claim to understand our expanded cone all that well.

Before we begin, we make some preliminary observations. Throughout the chapter we will let n be an arbitrary positive integer. Given n we define the set

$$\mathcal{I} = \{(i,j) : i = 1,\ldots,n-1, j = i+1,\ldots,n\}.$$

Assume C is some cone in the space $\mathcal{S}^n_+$. Then a linear inequality for C is of

the form

$$\sum_{i,j=1}^{n} a_{ij}x_{ij} \leq \alpha, \tag{3.1}$$

where $X = [x_{ij}]_{i,j=1}^{n} \in C$ is an arbitrary matrix. If (3.1) holds for all $X \in C$, it is valid for C. Now, since every matrix in C is symmetric, $x_{ij} = x_{ji}$ holds for all $(i,j) \in \mathcal{I}$. Then any inequality of the form (3.1) can be written as

$$\sum_{i=1}^{n} a_{ii}x_{ii} + \sum_{(i,j)\in\mathcal{I}} (a_{ij} + a_{ji})x_{ij} \leq \alpha.$$

Thus, when obtaining linear inequalities for a cone consisting of symmetric matrices it suffices to consider those where $a_{ji} = 0$ for all $(i,j) \in \mathcal{I}$. Sometimes it will be easier to write an inequality in terms of x_{ji} where $j > i$ rather than x_{ij}, we shall do this without comment – there is no problem as long as we don't use both x_{ij} and x_{ji} at the same time.

Another thing which is important to observe is that we are only looking for inequalities that are sometimes fulfilled with equality. If

$$\sum_{i=1}^{n} a_{ii}x_{ii} + \sum_{(i,j)\in\mathcal{I}} a_{ij}x_{ij} \leq \alpha$$

is a valid inequality for C we recall that the face it defines is the set

$$\left\{ \sum_{i=1}^{n} a_{ii}x_{ii} + \sum_{(i,j)\in\mathcal{I}} a_{ij}x_{ij} = \alpha : X \in C \right\}.$$

Then, if X is contained in the face defined by the inequality, we know that $X = \frac{1}{2}O + \frac{1}{2}2X$. Since we need the set to be a facet it must follow that both O and $2X$ are contained in the face. This is only fulfilled if $\alpha = 0$. We have altogether shown the following little proposition.

Proposition 3.1.1. *If $C \subseteq \mathcal{S}_+^n$ is a convex cone, any valid inequality for C that defines a face is of the form*

$$\sum_{i=1}^{n} a_{ii}x_{ii} + \sum_{(i,j)\in\mathcal{I}} a_{ij}x_{ij} \leq 0.$$

The coefficients a_{ii} for $i = 1, \ldots, n$ we will often refer to as the diagonal coefficients*, for the obvious reason.*

We can now move to our first order of business – determining the generators of $\mathcal{DD}_n$.

3.2 Generators of the diagonally dominant cone

Recall our definition of the cone $\mathcal{DD}_n$ in connection with Theorem 2.1.12, as well as the defining inequality set. At that time, we did not speak of the generators of this cone, but as mentioned this is what we turn to now. The result in this section is known – it follows directly from Proposition 1 in [8], but the proofs are our own, and serve partly to prove the result, partly to illustrate the technique we intend to apply in some other cases later.

Theorem 3.2.1. *Let n be a fixed positive integer, and define the following three sets:*

$$\begin{aligned} S_1 &= \{\mathbf{e}_i\mathbf{e}_i^T : i = 1, \ldots, n\}, \\ S_2 &= \{(\mathbf{e}_i + \mathbf{e}_j)(\mathbf{e}_i + \mathbf{e}_j)^T : (i,j) \in \mathcal{I}\}, \\ S &= S_1 \cup S_2. \end{aligned}$$

Then $\mathcal{DD}_n = \operatorname{cone} S$.

There are many ways of proving this – one is directly, by showing that all the matrices above are diagonally dominant, observing that diagonal dominance is retained when taking linear combinations, and then showing that any nonnegative diagonal dominant matrix can be decomposed as a sum of the matrices in S (as we did in the proof of Theorem 2.1.12). This approach is less useful to us here, however, as it is not so easy to generalise to a more complicated generator set. We will prove Theorem 3.2.1 in two ways. The first proof depends upon the fact that $\operatorname{cone} S$ is a simplicial cone, and is provided to illustrate the fundamentals of the proof strategy we use.

Proof using simplicial cone properties. Assume $X = [x_{ij}]_{i,j=1}^n$ is an arbitrary matrix in $\operatorname{cone} S$. According to 3.1.1 a valid face-defining inequality is then of the form

$$\sum_{i=1}^{n} a_{ii}x_{ii} + \sum_{(i,j)\in\mathcal{I}} a_{ij}x_{ij} \leq 0.$$

We consider what validity means as applied to the generators. First, from the generators in S_1 we see that for any $i = 1, \ldots, n$,

$$a_{ii} \leq 0.$$

From the generators in S_2 we see using symmetry that for $(i,j) \in \mathcal{I}$ the following must hold:

$$a_{ii} + a_{ij} + a_{jj} \leq 0.$$

This means that

$$a_{ij} \leq -a_{ii} - a_{jj}.$$

Now, we are looking for facet-defining inequalities. To be specific, we want the set of elements in $\operatorname{cone} S$ for which the inequality holds with equality to be a facet. S is a set of $\frac{n(n+1)}{2}$ linearly independent vectors, so $\operatorname{cone} S$ is simplicial, and this in turn means that the inequality must hold with equality in all but one of the generators, see Proposition 1.7.5. Having equality in a generator in S_1 implies that for some i we have

$$a_{ii} = 0.$$

Equality in a generator in S_2 implies that for some $(i,j) \in \mathcal{I}$ we have

$$a_{ij} = -a_{ii} - a_{jj}.$$

Since a_{ii} and a_{jj} are nonpositive, this means that either all three coefficients in this sum are zero or a_{ij} is positive.

We can now obtain all facet-defining inequalities simply by choosing which of the inequalities will *not* hold with equality and seeing what we get. First, let i be some fixed integer with $1 \leq i \leq n$. Assume $a_{ii} \neq 0$, which means $a_{ii} < 0$. Without loss of generality we can assume $a_{ii} = -1$. Now we must have equality in all the other generators, so no other diagonal coefficients can be nonzero. Further, for any $j > i$ we have that

$$a_{ii} + a_{ij} = 0 \Leftrightarrow a_{ij} = 1,$$

and for $j < i$ we similarly have that $a_{ji} = 1$. This satisfies all the requirements for being a facet-defining inequality, and so we know that we have found a facet-defining inequality

$$x_{ii} \geq \sum_{j=1}^{i-1} x_{ji} + \sum_{j=i+1}^{n} x_{ij}.$$

Recall that X is symmetric, so $x_{kl} = x_{lk}$ for all k, l. Using this we can write the inequality as

$$x_{ii} \geq \sum_{j=1, j\neq i}^{n} x_{ij}.$$

This works for any i, and we have found all the facet-defining inequalities we will find from the first type of generators.

Now, we consider what it would mean to have strict inequality for one of the generators in S_2. By the preceding discussion we know that all diagonal coefficients are zero, and we also know that their negatives bound the off-diagonal coefficients above, so the off-diagonal coefficients must now be nonpositive. Essentially, the only possible option is to choose some $(i, j) \in \mathcal{I}$ and let $a_{ij} < 0$, while leaving all other coefficients equal to zero. Since we can scale we can assume $a_{ij} = -1$, and we get the following facet-defining inequalities:

$$x_{ij} \geq 0 \text{ for } (i, j) \in \mathcal{I}.$$

These are all the facet-defining inequalities we obtain from the second set of generators, and since we have no further inequalities we have found them all. The inequalities that are facet-defining for cone S are then precisely the ones we used to define $\mathcal{DD}_n$ in Chapter 2, and so the theorem follows. □

We see that as long as we are working with a simplicial cone, finding the defining linear inequality system is straightforward. As such we could create a multitude of cones by replacing some generators in the generating set of $\mathcal{DD}_n$ in such a fashion that they remain linearly independent. It does not seem like any obvious candidates would be much more interesting than $\mathcal{DD}_n$, however, and we would rather expand the cone. In that case we cannot retain the simplicial cone property, so we step through the proof again without directly using this property, thus showing how we intend to obtain linear inequality systems for our expanded cones.

Alternate proof of Theorem 3.2.1. Assume $X = [x_{ij}]_{i,j=1}^{n}$ is an arbitrary matrix in cone S. According to 3.1.1 a valid face-defining inequality is then of the form

$$\sum_{i=1}^{n} a_{ii} x_{ii} + \sum_{(i,j)\in\mathcal{I}} a_{ij} x_{ij} \leq 0.$$

Like in the previous proof we can draw the conclusions

$$a_{ii} \leq 0 \text{ for } i = 1, \ldots, n.$$
$$a_{ij} \leq -a_{ii} - a_{jj} \text{ for } (i,j) \in \mathcal{I}.$$

Again, we observe that a facet must have dimension $\frac{n(n+1)}{2} - 1$. Observe that for $(i,j) \in \mathcal{I}$ we can set $a_{ij} < 0$ and let all the other coefficients be equal to zero. Scaling so $a_{ij} = -1$ we obtain the inequality

$$x_{ij} \geq 0.$$

This satisfies the conditions for being a valid inequality. Considering the generating set S we see immediately that this inequality is satisfied with inequality in all but one of the generators in S, and by inspection that set is linearly independent, and so the face defined by our inequality has dimension $\frac{n(n+1)}{2} - 1$, meaning it is a facet. Facets of this kind will be called *trivial facets*.

Now assume some linear inequality defines a nontrivial facet F. If it is the case for F that there is some $(i,j) \in \mathcal{I}$ such that $x_{ij} = 0$ for all $X \in F$, then F is contained within a trivial facet, and cannot be a facet. Thus, F must hold with equality in all the generators in S_2. This is our crucial observation, and it means that for all $(i,j) \in \mathcal{I}$ it must be true that

$$a_{ij} = -(a_{ii} + a_{jj}).$$

We see that this gives us a linear system of equalities which is already solved in terms of the free variables, which are the diagonal coefficients. Thus, the solution set is a subspace spanned by the vectors in which

$$a_{ii} = -1, a_{ij} = 1\ (j > i), a_{ji} = 1\ (j < i),$$

for $i = 1, \ldots, n$. As $a_{ii} \geq 0$ from the generators in S_1 any solution of the above inequalities is a nonnegative linear combination of the vectors illustrated above, meaning the system describes the cone entirely, and it is clear that it is also irredundant. This means that according to Theorem 1.7.1 we have found all the facets. □

3.3 Expanding $\mathcal{DD}_n$

We are armed with knowledge of the generators of $\mathcal{DD}_n$, and we can move on to the second part of our plan – adjoining more generators. There are two obvious ways in which to do this. The first is to add generators with more basis vector terms, the second is to change the weights. We will consider each in turn.

First, however, let us consider more carefully what it means to test for membership in a cone. A convex cone can be described in several ways, and we are describing our cones in terms of their generators and their facet-defining inequalities. If $\mathcal{W}$ is some set of k matrices, the following linear program can be used in determining membership of cone $\mathcal{W}$, assuming we are given some matrix X:

$$\begin{array}{rll} \text{minimise} & 0 & \\ \text{subject to} & X = \sum_{i=1}^{k} \lambda_i W_i & \\ & \lambda_i \geq 0 \quad \text{for} \quad i = 1, \ldots, k & \end{array} \tag{3.2}$$

As mentioned in Section 1.9, there exist efficient (in the big-O sense) algorithms to solve this problem – linear programming algorithms will determine whether or not the problem has a solution at all, which is equivalent to $X \in \operatorname{cone} \mathcal{W}$. This test, however, does not provide much insight into how a cone "looks" as a geometric object, and perhaps a system of inequalities, as is the case with $\mathcal{DD}_n$, might sometimes prove easier to compute. It certainly seems worthwhile to at least try. Presently we will consider some different strategies as far as expanding the cone $\mathcal{DD}_n$ with more generators go, and see whether we get a nice system or not.

A computer program we will be using is PORTA, a program written by Thomas Christof and Andreas Loebel[1]. PORTA is a program that converts representations of convex sets from a representation in terms of their generators to a representation in terms of inequalities (i.e. halfspaces) and back – internally it uses some sort of Fourier-Motzkin algorithm-based approach, but we are not really concerned with how it works. PORTA does not work well for high-dimensional cones (in our case, we are looking at cones of matrices, and beyond matrices of about order 8 it rarely manages to deal with the systems), but we will use it to generate working hypotheses about how a cone's defining system of inequalities is structured, and then see if we can prove them.

3.4 Triple diagonal cone

We are generalising the diagonally dominant cone. One fairly obvious way of doing this is by adding a term generated by three vectors, that is, by including the following set among the generators:

$$\{(\mathbf{e}_i+\mathbf{e}_j+\mathbf{e}_k)(\mathbf{e}_i+\mathbf{e}_j+\mathbf{e}_k)^T : i = 1,\dots,n-2, j = i+1,\dots,n-1, k = j+1,\dots,n\}$$

Unfortunately, when investigating with PORTA, this set turns out to have ever-increasing complexity in its inequalities, and not in a very nice way. We will not consider this cone further in the thesis.

Instead we try to see if we can gain something from limiting the generator set somewhat. Instead of including the entire set just described, we pick only the "triple" elements that are right on the diagonal. We state explicitly what we mean.

Definition 3.4.1. *Let*

$$\begin{aligned} S_1 &= \{\mathbf{e}_i\mathbf{e}_i^T, i = 1,\dots,n\},\\ S_2 &= \{(\mathbf{e}_i+\mathbf{e}_j)(\mathbf{e}_i+\mathbf{e}_j)^T, i = 1,\dots,n-1, j = i+1,\dots,n\},\\ S_3 &= \{(\mathbf{e}_i+\mathbf{e}_{i+1}+\mathbf{e}_{i+2})(\mathbf{e}_i+\mathbf{e}_{i+1}+\mathbf{e}_{i+2})^T : i = 1,\dots,n-2\},\\ S &= S_1 \cup S_2 \cup S_3. \end{aligned}$$

Then $C = \operatorname{cone} S$ *shall be called the* big triple diagonal cone.

We see that this cone is generated by a subset of the generators in our first attempt at a definition, so we might at least hope that this turns out to be simpler.

[1] At the time of this writing, it was available at http://www2.iwr.uni-heidelberg.de/groups/comopt/software/PORTA/.

When using PORTA to observe some inequality sets for this cone, the system turns out to be quite complex still. Before moving on, then, we make one more assumption that might make analysis easier. Assume $X \in C$. Then if $x_{ij} > 0$ for $j > i+2$, it must follow that a generator in S_2 was used to cover that particular element, and as there is only one such generator for each such element, then we know what its weight must be when writing X as a combination of generators. Therefore, we can define

$$X' = X - \sum_{(i,j),j>i+2} x_{ij}(\mathbf{e}_i + \mathbf{e}_j)(\mathbf{e}_i + \mathbf{e}_j)^T.$$

The only nonzero elements in X' are those for which $|i-j| < 3$. We define the following cone.

Definition 3.4.2. *Let*

$$\begin{aligned}
S_1 &= \{\mathbf{e}_i\mathbf{e}_i^T, i = 1, \ldots, n\},\\
S_2 &= \{(\mathbf{e}_i + \mathbf{e}_{i+1})(\mathbf{e}_i + \mathbf{e}_{i+1})^T, i = 1, \ldots, n-1\},\\
S_3 &= \{(\mathbf{e}_i + \mathbf{e}_{i+2})(\mathbf{e}_i + \mathbf{e}_{i+2})^T, i = 1, \ldots, n-1\},\\
S_4 &= \{(\mathbf{e}_i + \mathbf{e}_{i+1} + \mathbf{e}_{i+2})(\mathbf{e}_i + \mathbf{e}_{i+1} + \mathbf{e}_{i+2})^T : i = 1, \ldots, n-2\},\\
S &= S_1 \cup S_2 \cup S_3 \cup S_4.
\end{aligned}$$

Then we define the triple diagonal cone $\mathcal{TD}_n = \operatorname{cone} S$.

We see that X is in the big triple diagonal cone if and only if X' is in the triple diagonal cone, so we will consider the inequality systems generated for the triple diagonal cone. It is clearly not a simplicial cone, and we expect it to be more complicated than $\mathcal{DD}_n$. We consider some examples generated using PORTA. Note that here we assume implicitly that $x_{ij} = 0$ if $j > i+2$.

3.4.1 Examples

If $n = 2$, the cone we describe is just $\mathcal{DD}_2$, so we ignore that trivial case, and begin by considering the cone $\mathcal{TD}_3$. Note that there are quite a few inequalities. We have written the PORTA-generated output as succinctly as possible, hopefully without becoming overly obtuse.

1. $x_{ij} \geq 0$ for $i = 1, 2$, $j = i+1, \ldots, 3$.
2. $x_{11} \geq x_{12}, \qquad x_{11} \geq x_{13}$.
3. $x_{22} \geq x_{12}, \qquad x_{22} \geq x_{23}$.
4. $x_{33} \geq x_{13}, \qquad x_{33} \geq x_{23}$.
5. $x_{11} + x_{23} \geq x_{12} + x_{13}$.
6. $x_{22} + x_{13} \geq x_{12} + x_{23}$.
7. $x_{33} + x_{12} \geq x_{13} + x_{23}$.

Thus far, the pattern looks clean and simple, but we must consider the cone also for larger n in order to get a good feel for it. We look at $\mathcal{TD}_4$:

1. $x_{ij} \geq 0$ for $i = 1, \ldots, 3$, $j = i+1, i+2$.
2. $x_{11} \geq x_{12}, \qquad x_{11} \geq x_{13}$.
3. $x_{22} \geq x_{23}$.
4. $x_{33} \geq x_{23}$.
5. $x_{44} \geq x_{23}, \qquad x_{44} \geq x_{24}$.
6. $x_{22} \geq x_{12} + x_{13}$.
7. $x_{33} \geq x_{13} + x_{34}$.
8. $x_{11} + x_{23} \geq x_{12} + x_{13}$.
9. $x_{22} + x_{13} \geq x_{12} + x_{23}$.
10. $x_{22} + x_{34} \geq x_{23} + x_{34}$.
11. $x_{33} + x_{12} \geq x_{13} + x_{23}$.
12. $x_{33} + x_{24} \geq x_{23} + x_{34}$.
13. $x_{44} + x_{23} \geq x_{24} + x_{34}$.
14. $x_{22} + x_{13} + x_{34} \geq x_{12} + x_{23} + x_{24}$.
15. $x_{33} + x_{12} + x_{24} \geq x_{13} + x_{23} + x_{34}$.
16. $x_{11} + x_{44} + x_{23} \geq x_{12} + x_{13} + x_{24} + x_{34}$.

Some distinctions become apparent here. Consider the inequalities 6 and 7, which have no analogue for x_{11} and x_{44}, as well as the set of inequalities 8-13, which exhibits the same sort of different treatment of certain diagonal elements. The reason is that in $\mathcal{TD}_n$, all elements are not equal: Every diagonal element is covered by one generator in S_1, one in S_2 and one in S_3. However, x_{11} and x_{nn} are in general covered by one generator in S_4, while x_{22} and $x_{n-1,n-1}$ are covered by two. Here, we have no more diagonal elements, but in general $x_{33}, \ldots, x_{n-2,n-2}$ are covered by three elements from S_4. Since the elements of the matrices are different in this sense, the system is much more complicated than it might be otherwise – for instance, when analysing $\mathcal{DD}_n$ we saw that all diagonal elements were "equal" in terms of the generators covering them, as were the off-diagonal ones.

Finally, $\mathcal{TD}_5$, in which x_{33} is one of those diagonal elements which will dominate when n increases – the kind covered by three generators in S_4:

1. $x_{ij} \geq 0$ for $i = 1, \ldots, 4$, $j = i+1, i+2$.
2. $x_{11} \geq x_{12}, \qquad x_{11} \geq x_{13}$.
3. $x_{22} \geq x_{23}$.
4. $x_{44} \geq x_{34}$.
5. $x_{55} \geq x_{35}, \qquad x_{55} \geq x_{45}$.
6. $x_{22} \geq x_{12} + x_{24}$.

7. $x_{33} \geq x_{13} + x_{34}$.

8. $x_{33} \geq x_{13} + x_{35}$.

9. $x_{44} \geq x_{24} + x_{45}$.

10. $x_{11} + x_{23} \geq x_{12} + x_{13}$.

11. $x_{22} + x_{13} \geq x_{12} + x_{13}$.

12. $x_{22} + x_{34} \geq x_{23} + x_{24}$.

13. $x_{33} + x_{24} \geq x_{23} + x_{34}$.

14. $x_{44} + x_{23} \geq x_{24} + x_{34}$.

15. $x_{44} + x_{35} \geq x_{34} + x_{45}$.

16. $x_{55} + x_{34} \geq x_{35} + x_{45}$.

17. $x_{22} + x_{13} + x_{34} \geq x_{12} + x_{23} + x_{24}$.

18. $x_{33} + x_{12} + x_{24} \geq x_{13} + x_{23} + x_{34}$.

19. $x_{33} + x_{24} + x_{45} \geq x_{23} + x_{34} + x_{35}$.

20. $x_{44} + x_{23} + x_{35} \geq x_{24} + x_{34} + x_{45}$.

21. $x_{11} + x_{44} + x_{23} \geq x_{12} + x_{13} + x_{24} + x_{34}$.

22. $x_{22} + x_{55} + x_{24} \geq x_{23} + x_{24} + x_{35} + x_{45}$.

23. $x_{33} + x_{24} + x_{12} + x_{45} \geq x_{13} + x_{23} + x_{34} + x_{35}$.

24. $x_{11} + x_{44} + x_{23} + x_{34} \geq x_{12} + x_{13} + x_{24} + x_{34} + x_{45}$.

25. $x_{22} + x_{55} + x_{13} + x_{34} \geq x_{12} + x_{23} + x_{24} + x_{35} + x_{45}$.

We are starting to see that there is some structure to the system. It seems that most of the inequalities that are facet-defining and valid for the elements at one order stay for the next, but there is a definite sense that the new ones in each step are more complicated than they were last time. The number of inequalities for small n is indicated in the following little table:

n	Number of inequalities
3	12
4	22
5	36
6	54
7	75
8	99

As we see, there is a definite tendency towards increasing growth here, and as we saw above not only does the number of inequalities grow, but their complexity grows as well. If we tried to find all facet-defining inequalities much as we tried to find the ones for $\mathcal{DD}_n$ in the second proof of 3.2.1 we would easily find that the trivial facets are the same as in that case, and that for a general facet we must have

$$\begin{gathered} a_{ii} \leq 0 \text{ for } i = 1, \ldots, n, \\ a_{ij} \leq -a_{ii} - a_{jj} \text{ for } i = 1, \ldots, n-1, j = i+1, i+2, j \leq n, \\ a_{i,i+1} + a_{i,i+2} + a_{i+1,i+2} \leq -a_{ii} - a_{i+1,i+1} - a_{i+2,i+2} \text{ for } i = 1, \ldots, n-2. \end{gathered}$$

Further we must have equality for at least one generator with $x_{ij} > 0$ for $j > i$. This corresponds to equality in one of the two last types of inequalities above. However, choosing which will hold is very complex, and depends on whether we choose to include generators in $S_2 \cup S_3$ or S_4 in our facet, and to make a long story short we have been unable to come up with a nice way of structuring the proof – the number of possibilities balloons quickly, and our exhibited inequalities do not quite suggest any tempting theories as to the structure of the set of facets. We shall instead turn our attention to another manner in which $\mathcal{DD}_n$ can be expanded, hoping that it will prove more tractable.

3.5 Another expansion

Our last approach ran into some difficulties, and one in particular was that the number of generators per element was not constant, which ultimately had its roots in the fact that some generators had supports of different sizes than the others. One way of expanding the diagonally dominant cone and keeping to generators with nice support is to define the following set:

$$\mathcal{C}_k = \text{cone}\{(r\mathbf{e}_i + s\mathbf{e}_j)(r\mathbf{e}_i + s\mathbf{e}_j)^T : (i,j) \in \mathcal{I}, r \in \mathbb{N}, s \in \mathbb{N}, r + s \leq k\}$$

With $k = 2$, this is simply $\mathcal{DD}_n$. We try to set $k = 3$ and see what we can prove. Using PORTA again we obtain the following explicit systems, which will suggest to us a hypothesis.

3.5.1 Examples

The cone when $n = 2$ is determined by the following inequalities:

1. $x_{12} \geq 0$.
2. $2x_{ii} \geq x_{12}$ for $i = 1, 2$.
3. $2x_{ii} + x_{jj} \geq 3x_{12}$ for $i, j \in \{1, 2\}$ with $i \neq j$.

When $n = 3$, the basic inequalities are

1. $x_{ij} \geq 0$ for $i = 1, 2$ and $j = i+1, \ldots, 3$.
2. $2x_{ii} \geq \sum_{k=1, k\neq i}^{3} x_{ik}$ for $i = 1, \ldots, 3$.

For all choices of $i, j, k \in \{1, 2, 3\}$ and i, j, k different, the following inequalities also play a part in determining the cone:

3. $4x_{ii} + 2x_{jj} \geq 6x_{ij} + 2x_{ik} + x_{jk}$.
4. $2x_{ii} + x_{jj} + x_{kk} \geq 3x_{ij} + 3x_{ik} + 2x_{jk}$.
5. $2x_{ii} + 2x_{jj} + x_{kk} \geq 4x_{ij} + 3x_{ik} + 3x_{jk}$.
6. $4x_{ii} + 2x_{jj} + x_{kk} \geq 6x_{ij} + 4x_{ik} + 3x_{jk}$.

When $n = 4$ things start to get more complicated. The two most basic kinds of inequalities here are

1. $x_{ij} \geq 0$ for $i = 1, \ldots, 4$ and $j = i + 1, \ldots, n$.
2. $2x_{ii} \geq \sum_{k=1, k\neq i}^{4} x_{ik}$ for $i = 1, \ldots, 4$.

For all choices of $i, j, k, l \in \{1, 2, 3, 4\}$ with i, j, k, l different, the following inequalities are also valid:

3. $4x_{ii} + 2x_{jj} \geq 6x_{ij} + 2x_{ik} + 2x_{il} + x_{jk} + x_{jl}$.
4. $4x_{ii} + 2x_{jj} + 2x_{kk} \geq 6x_{ij} + 6x_{ik} + 4x_{jk} + 2x_{il} + x_{jl} + x_{kl}$.
5. $4x_{ii} + 4x_{jj} + 2x_{kk} \geq 8x_{ij} + 6x_{ik} + 6x_{jk} + 2x_{il} + 2x_{jl} + x_{kl}$.
6. $2x_{ii} + x_{jj} + x_{kk} + x_{ll} \geq 3x_{ij} + 3x_{ik} + 3x_{il} + 2x_{jk} + 2x_{jl} + 2x_{kl}$.
7. $2x_{ii} + 2x_{jj} + x_{kk} + x_{ll} \geq 4x_{ij} + 3x_{ik} + 3x_{il} + 3x_{jk} + 3x_{jl} + 2x_{kl}$.
8. $2x_{ii} + 2x_{jj} + 2x_{kk} + x_{ll} \geq 4x_{ij} + 4x_{ik} + 3x_{il} + 4x_{jk} + 3x_{jl} + 3x_{kl}$.
9. $4x_{ii} + 2x_{jj} + x_{kk} + x_{ll} \geq 6x_{ij} + 4x_{ik} + 4x_{il} + 3x_{jk} + 3x_{jl} + 2x_{kl}$.
10. $4x_{ii} + 2x_{jj} + 2x_{kk} + x_{ll} \geq 6x_{ij} + 6x_{ik} + 4x_{il} + 4x_{jk} + 3x_{jl} + 3x_{kl}$.
11. $4x_{ii} + 4x_{jj} + 2x_{kk} + x_{ll} \geq 8x_{ij} + 6x_{ik} + 6x_{jk} + 4x_{il} + 4x_{jl} + 3x_{kl}$.
12. $8x_{ii} + 4x_{jj} + 2x_{kk} + x_{ll} \geq 12x_{ij} + 8x_{ik} + 6x_{il} + 6x_{jk} + 4x_{jl} + 3x_{kl}$.
13. $8x_{ii} + 4x_{jj} + 2x_{kk} \geq 12x_{ik} + 8x_{ik} + 6x_{jk} + 4x_{il} + 2x_{jl} + x_{kl}$.

Note above that PORTA works pretty hard to make all the constants be integers.

Our first observation is that we have far more inequalities here. PORTA gives up beyond $n = 5$, but we sum up the number of inequalities in the following little table:

n	Number of inequalities
2	5
3	24
4	144
5	1065

This is a very large number of inequalities, and so it seems unlikely that we shall gain a more efficient test in the $O(g(n))$ sense than the one linear programming techniques could furnish us with, but we might still obtain insight into the structure of the cone by finding the linear system describing it, and so we shall soldier on. While this system is in general much larger than the one describing the previous cone it is much simpler in the way it treats its variables, and so we hope to find something here.

3.5.2 Analysis

The key insight when analysing the systems we have just seen is that the diagonal coefficients are all powers of 2 – and since we can scale the inequalities without losing anything, we can go farther: the smallest diagonal coefficient can be assumed to be 1, and then all the other diagonal coefficients take their values in a set of consecutive powers of 2 starting with 1. There also seems to be some relation between the diagonal coefficients and the coefficients of the off-diagonal elements as well, though it is not, on the surface of it, as clear.

This system has many of the features we were missing in the last one, and we will be more successful in our analysis here. The proof is rather long, so we have tried to leave out those details which are easy to see while retaining the main thrust of the arguments. First, we need a definition.

Definition 3.5.1 (Doubling chains)**.** *Let S be some vector of k real numbers. We shall call S a* doubling chain *if there exists an ordering $\gamma_1, \ldots, \gamma_k$ of the elements of S such that*

$$\gamma_j = 2^{p_j}\gamma_1 \text{ for } j = 2, \ldots, k,$$

where p_j is either equal to p_{j-1} or $p_{j-1}+1$, $p_1 = 0$ and $p_k \geq p_1 + 1$. A set of one element is obviously a doubling chain, and we shall refer to these as trivial doubling chains.

Examples of doubling chains include: $(1, 2, 4, 8)$, $(1, 2, 2, 2)$ and $(2, 4, 4, 8)$. In particular, note that in our exhibited inequalities earlier the nonzero diagonal coefficients always form a single doubling chain. This is what we are aiming to prove: That the facet-defining inequalities are derived by setting the diagonal coefficients either to zero or to some doubling chain, and choosing the off-diagonal coefficients in some sensible manner, where we shall give meaning to "sensible" later.

Now, let $n \in \mathbb{N}$ be given. We remind the reader that we have defined $\mathcal{I} = \{(i, j) : i = 1, \ldots, n-1, j = i+1, \ldots, n\}$. We define the four sets

- $S_1 = \{\mathbf{e}_i\mathbf{e}_i^T : i = 1, \ldots, n\}$,
- $S_2 = \{(\mathbf{e}_i + \mathbf{e}_j)(\mathbf{e}_i + \mathbf{e}_j)^T : (i, j) \in \mathcal{I}\}$,
- $S_3 = \{(\mathbf{e}_i + 2\mathbf{e}_j)(\mathbf{e}_i + 2\mathbf{e}_j)^T : (i, j) \in \mathcal{I}\}$,
- $S_4 = \{(\mathbf{2}e_i + \mathbf{e}_j)(\mathbf{2}e_i + \mathbf{e}_j)^T : (i, j) \in \mathcal{I}\}$,

and we let $\mathcal{C}_3 = \text{cone}(S_1 \cup S_2 \cup S_3 \cup S_4)$. From Proposition 3.1.1 we know that any candidate for a facet-defining inequality can be written on the form

$$\sum_{i=1}^{n} a_{ii}x_{ii} + \sum_{(i,j)\in\mathcal{I}} a_{ij}x_{ij} \leq 0$$

By virtue of being a superset of $\mathcal{DD}_n$, $\mathcal{C}_3$ is a full-dimensional cone in the space of $n \times n$ symmetric matrices. This means, see Theorems 1.7.1-1.7.3, that if we can find all the facet-defining inequalities we will have found the unique minimal system of inequalities defining of $\mathcal{C}_3$.

Validity of an inequality means, in particular, that it is valid in the various generators. We can use this to conclude that the following relations must hold for the coefficients in a valid inequality.

Validity in S_1: $a_{ii} \leq 0$ for $i = 1, \ldots, n$.

Validity in S_2: $a_{ij} \leq -a_{ii} - a_{jj}$ for $(i,j) \in \mathcal{I}$.

Validity in S_3: $a_{ij} \leq \frac{1}{2}(-a_{ii} - 4a_{jj})$ for $(i,j) \in \mathcal{I}$.

Validity in S_4: $a_{ij} \leq \frac{1}{2}(-4a_{ii} - a_{jj})$ for $(i,j) \in \mathcal{I}$.

Of course, by linearity an inequality that holds in the generators must also hold for the entire cone, so this is a necessary and sufficient criterion for being a valid inequality. A generator is obviously included in the face defined by such an inequality if and only if its corresponding constraint above holds with equality.

We have thus far obtained several pieces of information about our facet-defining inequalities and our cone. We summarise our findings in a little lemma for easy reference.

Lemma 3.5.2. *C_3 is a convex cone of dimension $\frac{n(n+1)}{2}$, so facets of C_3 have dimension $\frac{n(n+1)}{2} - 1$. Furthermore, any face-defining inequality for $\mathcal{C}_3$ is of the form*

$$\sum_{i=1}^{n} a_{ii}x_{ii} + \sum_{(i,j)\in\mathcal{I}} a_{ij}x_{ij} \leq 0, \tag{3.3}$$

where $X \in C_3$. Concerning the coefficients in (3.3) *we know the following four facts:*

$$\begin{aligned} &a_{ii} \leq 0 \text{ for } i = 1, \ldots, n. \\ &a_{ij} \leq -a_{ii} - a_{jj} \text{ for } (i,j) \in \mathcal{I}. \\ &a_{ij} \leq \frac{1}{2}(-a_{ii} - 4a_{jj}) \text{ for } (i,j) \in \mathcal{I}. \\ &a_{ij} \leq \frac{1}{2}(-4a_{ii} - a_{jj}) \text{ for } (i,j) \in \mathcal{I}. \end{aligned} \tag{3.4}$$

Furthermore, the face F' defined by a face-defining inequality is the convex cone generated by the generators whose corresponding inequalities in (3.4) *hold with equality.*

So far we have spoken much about facets, but not actually found any. We rectify this presently. Consider the following inequality, where $(i,j) \in \mathcal{I}$:

$$x_{ij} \geq 0. \tag{3.5}$$

The corresponding face is the set $\{X \in \mathcal{C}_3 : x_{ij} = 0\}$. Considering the generators in $S_1 \cup S_2$ contained in this set, it is clear that it has dimension at least $\frac{n(n+1)}{2} - 1$, and it is equally clear that it cannot have higher dimension (there are not enough nonzero elements in a matrix contained in this facet). From this we see that we have found a facet, and in future we will refer to facets of the type (3.5) as *trivial facets*.

Now assume we have some face-defining inequality with $a_{ii} = 0$ for $i = 1, \ldots, n$. Then for all $(i,j) \in \mathcal{I}$, $a_{ij} \leq 0$, and we see that any such inequality is implied by the trivial facet-defining inequalities. We have thus found all valid inequalities for $\mathcal{C}_3$ with all diagonal coefficients zero.

When we wish to move on to inequalities in which the diagonal coefficients are nonzero we need to know which of the three bounds on a_{ij} in Lemma 3.5.2

is the tightest. We show this in all generality (Keep in mind that as $a_{ii} \leq 0$ for all i, $-a_{ii} \geq 0$).

Lemma 3.5.3. *Assume $\alpha, \beta \in \mathbb{R}_+$. We consider the following three numbers computed from them:*

$$\frac{1}{2}(4\alpha + \beta), \alpha + \beta, \frac{1}{2}(\alpha + 4\beta).$$

Out of these three, the smallest one is

- $\frac{1}{2}(4\alpha + \beta)$ *if* $\alpha \leq \frac{1}{2}\beta$.
- $\alpha + \beta$ *if* $\frac{1}{2}\beta \leq \alpha \leq 2\beta$.
- $\frac{1}{2}(\alpha + 4\beta)$ *if* $2\beta \leq \alpha$.

Note that if $\alpha = 2\beta$ or $\alpha = \frac{1}{2}\beta$ two of the numbers attain the minimum at the same time.

Proof. Assume $\frac{1}{2}(4\alpha + \beta)$ is the smallest. This means that

$$\frac{1}{2}(4\alpha + \beta) \leq \alpha + \beta, \qquad \frac{1}{2}(4\alpha + \beta) \leq \frac{1}{2}(\alpha + 4\beta).$$

Sorting both of those, we see that they are equivalent to

$$\alpha \leq \frac{1}{2}\beta, \qquad \alpha \leq \beta.$$

The second inequality is implied by the first, since $\alpha, \beta \geq 0$. The necessary condition $\alpha \leq \frac{1}{2}\beta$ is also sufficient.

Assume $\alpha + \beta$ is the smallest. This means that

$$\alpha + \beta \leq \frac{1}{2}(4\alpha + \beta), \qquad \alpha + \beta \leq \frac{1}{2}(\alpha + 4\beta).$$

Sorting again, we see that these are equivalent to

$$\frac{1}{2}\beta \leq \alpha, \qquad \alpha \leq 2\beta.$$

Again these necessary conditions are also sufficient.

Finally, assume $\frac{1}{2}(\alpha + 4\beta)$ is the smallest. This means that

$$\frac{1}{2}(\alpha + 4\beta) \leq \frac{1}{2}(4\alpha + \beta), \qquad \frac{1}{2}(\alpha + 4\beta) \leq \alpha + \beta.$$

Sorting this out, we see that we must have

$$\beta \leq \alpha, \qquad 2\beta \leq \alpha.$$

The first condition is implied by the second, and again the necessary condition $2\beta \leq \alpha$ is also sufficient. □

Now we can at last get to work on facet-defining inequalities with nonzero diagonal coefficients. Assume F is a facet defined by some inequality

$$\sum_{i=1}^{n} a_{ii}x_{ii} + \sum_{(i,j)\in\mathcal{I}} a_{ij}x_{ij} \leq 0.$$

If, for some $(i,j) \in \mathcal{I}$, F contains no generator with $x_{ij} \neq 0$, then $x_{ij} = 0$ is a true equality for the set F. Accordingly, F is contained within the corresponding trivial facet, and as such is not a facet. Therefore we know F must contain some generator with $x_{ij} > 0$. This means that (at least) one of the bounds on a_{ij} in Lemma 3.5.2 must hold with equality, and so we are justified in setting a_{ij} to be equal to the smallest of the three right-hand sides. Using Lemma 3.5.3 to determine which that is, we obtain the following corollary.

Corollary 3.5.4 (Off-diagonal coefficients in facets)**.** *Assume we have a facet-defining inequality*

$$\sum_{i=1}^{n} a_{ii}x_{ii} + \sum_{(i,j)\in\mathcal{I}} (a_{ij} + a_{ji})x_{ij} \leq 0.$$

Then the following must be true for all $(i,j) \in \mathcal{I}$:

$$a_{ij} = \min\left\{\frac{1}{2}(-4a_{ii} - a_{jj}), -a_{ii} - a_{jj}, \frac{1}{2}(-a_{ii} - 4a_{jj})\right\}. \tag{3.6}$$

The minimum is

- $\frac{1}{2}(-4a_{ii} - a_{jj})$ *if* $-a_{ii} \leq -\frac{1}{2}a_{jj}$.
- $-a_{ii} - a_{jj}$ *if* $-\frac{1}{2}a_{jj} \leq -a_{ii} \leq -2a_{jj}$.
- $\frac{1}{2}(-a_{ii} - 4a_{jj})$ *if* $-2a_{jj} \leq -a_{ii}$.

As such, *nontrivial facet-defining inequalities are uniquely determined by the coefficients of the diagonal elements of X*. This will make things a lot easier for us[2].

It is time to find more facets. It will be easier to also treat the case of only one nonzero diagonal coefficient by itself. Assume for some i that $a_{ii} < 0$, and that $a_{jj} = 0$ for all $j \neq i$. Let $(i',j') \in \mathcal{I}$. Since we are looking for facets, Corollary 3.5.4 applies, and all the nondiagonal coefficients are uniquely determined (they are $-\frac{1}{2}a_{ii}$ if either $i' = i$ or $j' = i$ and zero otherwise).

So what is the dimension of the determined face? Well, the face contains every generator in S_1 except $\mathbf{e}_i\mathbf{e}_i^T$, and every a_{ij} is chosen so as to provide at least one generator with $x_{ij} \neq 0$ – it is clear that all of these are linearly independent as their supports do not overlap, so we have at a minimum $\frac{n(n+1)}{2} - 1$ linearly independent generators, meaning that the dimension is at least this much, and since the face cannot have larger dimension this must in fact be the dimension of the face, meaning it is a facet. Given that facet-defining inequalities with at least one nonzero diagonal coefficient are uniquely determined by the coefficients

[2]This did not hold in the triple diagonal cone.

on the diagonal by Corollary 3.5.4, there are no other facet-defining inequalities with just one nonzero diagonal coefficient.

Now we can at last consider a general facet-defining inequality. It is time to consider facet-defining inequalities with more than one nonzero coefficient on the diagonal. Henceforth, all off-diagonal coefficients are assumed to be chosen in accordance with Corollary 3.5.4. If not, they cannot possibly be facets and as such they are not interesting.

We wish to show that in any such inequality, the nonzero coefficients form a doubling chain. The basic idea is to take a face-defining inequality in which they do not and show that all the generators of that face are contained within a face defined by an inequality in which they do. This shows that the second face contains the first, and so only the second can possibly be a facet. The important point in the proofs to come is that the generators (excepting those in S_1) contained in a face where the off-diagonal coefficients are chosen in this way obviously depend entirely upon which minimum is attained in (3.6). As such, when we go from an inequality in which the coefficients are not a doubling chain to one in which they are we only need to show that the minima attained in (3.6) do not change, and we are done.

Lemma 3.5.5. *Let F' be a face defined by an inequality*

$$\sum_{i=1}^{n} a_{ii}x_{ii} + \sum_{(i,j)\in\mathcal{I}} a_{ij}x_{ij} \leq 0,$$

where at least two of the coefficients on the diagonal are nonzero. If there exists some a_{ii} such that

$$-a_{ii} \neq -\frac{1}{2}a_{jj}, -a_{ii} \neq -2a_{jj}$$

are true for all $j \neq i$, F' is not a facet.

Proof. Let i be as stated, and define the set

$$M = \left\{-\frac{1}{2}a_{jj} : j \neq i\right\} \cup \{-2a_{jj} : j \neq i\}.$$

Assume the set $\{x \in M : x < -a_{ii}\}$ is nonempty. Let

$$\alpha = \max\{x \in M : x < -a_{ii}\}$$

Now consider the face-defining inequality obtained by setting $\widetilde{a_{jj}} = a_{jj}$ for $j \neq i$, $\widetilde{a_{ii}} = \alpha$ and choosing the off-diagonal coefficients according to Corollary 3.5.4. We call the face defined by this new equality F. We propose that $F' \subseteq F$. Since we have changed the relationship only between a_{ii} and other diagonal elements, we know that only the generators involving $\mathbf{e}_i$ can have changed from F' to F. Now, let $k \neq i$.

- If $-a_{ii} < -\frac{1}{2}a_{kk}$, then $\alpha < -\frac{1}{2}a_{kk}$, and nothing has changed.
- If $-a_{ii} > -\frac{1}{2}a_{kk}$ and $\alpha < -\frac{1}{2}a_{kk}$, then $-\frac{1}{2}a_{kk} \in \{x \in M : x < -a_{ii}\}$, and the assumed relation contradicts our choice of α as the maximum of the set.

- If $-a_{ii} > -2a_{kk}$ and $\alpha < -2a_{kk}$, then $-2a_{kk} \in \{x \in M : x < -a_{ii}\}$, and the assumed relation contradicts our choice of α as the maximum of the set.

We conclude that every generator which is contained in F' is also contained in F. Further, since $-a_{ii}$ is now either double or half of another diagonal coefficient we have more generators in F than F'. This means that F' cannot possibly be a facet.

Now, if $\{x \in M : x < -a_{ii}\}$ is empty, it should be clear that we can instead define the set $\{x \in M : x > -a_{ii}\}$ and perform the proof in an entirely similar fashion choosing the minimum of this set instead, so we are done. □

A different formulation of Lemma 3.5.5 is that in a facet-defining inequality, every diagonal coefficient is contained in some doubling chain composed of diagonal coefficients. We are almost there – our initial conjecture was that they are all contained in a single doubling chain. We prove that if they are not we do not have a facet. The next proof uses the same strategy once again.

Lemma 3.5.6. *Assume we have a face-defining inequality F' in which not all the diagonal coefficients are in a single doubling chain. Then F' is not a facet.*

Proof. Assume $-a_{ii}$ is the smallest diagonal coefficient in absolute value. From Lemma 3.5.5 we know $-a_{ii}$ is contained in a doubling chain, call it D. Let R be the set containing all diagonal coefficients not contained in D. Then R consists of one or more doubling chains, call them $D_1, \ldots, D_k$. Let

$$M = \left\{\frac{1}{2}x : x \in D\right\} \cup \{2x : x \in D\}.$$

For every D_1 let d_j be the smallest element in the doubling chain, and let $\alpha_j = \max\{x \in M : x < d_j\}$. Since we assumed that $-a_{ii}$ is the smallest diagonal coefficient α_j is well-defined. Clearly it is the case that for all j there exists some $\beta_j > 1$ such that $d_j = \beta_j \alpha_j$.

Now let

$$\beta = \min_j\{\beta_j\}.$$

Let F be the face defined by the inequality obtained from the inequality defining F' by scaling every coefficient in R by $\frac{1}{\beta}$ while still picking the off-diagonal coefficients according to Corollary 3.5.4. We need to show that we have lost no generators.

Assume $-a_{kk} \in R$. Then $-a_{kk}$ is in some doubling chain, say D_j. Then there is some m' such that $-a_{kk} = 2^{m'} d_j$. Since $d_j = \beta_j \alpha_j$ it follows that $-a_{kk} = 2^{m'} \beta_j \alpha_j$. Since $\alpha_j \in D$, there is some m such that $-a_{kk} = -2^m \beta_j a_{ii}$. We observe that by the definition of β, $\frac{\beta_j}{\beta} \geq 1$ (the fraction equals one if D_j is the chain for which the minimum was attained when choosing β. Let $-a_{tt}$ be an arbitrary element in D. Then there is some r such that $-a_{tt} = 2^r a_{ii}$.

- If $-a_{kk} < -\frac{1}{2}a_{tt}$, then $-\frac{1}{\beta}a_{kk} < -a_{kk} < -\frac{1}{2}a_{tt}$, and nothing has changed.
- If $-a_{kk} > -\frac{1}{2}a_{tt}$, then $m \geq r-1$, and since $-\frac{1}{\beta}a_{kk} = -\frac{\beta_j}{\beta}2^m a_{ii} \geq -2^m a_{ii}$ it follows that $-\frac{1}{\beta}a_{kk} \geq -\frac{1}{2}a_{tt}$.

- If $-a_{kk} > -2a_{tt}$, then $m \geq r+1$, and since $-\frac{1}{\beta}a_{kk} = -\frac{\beta_j}{\beta}2^m a_{ii} \geq -2^m a_{ii}$, $-\frac{1}{\beta}a_{kk} \geq -2a_{tt}$.

We have shown that whichever minimum was used to choose the coefficient a_{kt} in the inequality defining F' still holds with equality when making the same choice in our new inequality, and so F must contain all the generators contained in F', and the result follows. □

So far we have shown that we can assume all the facet-defining inequalities are either the trivial ones, the uniquely determined ones with only one nonzero diagonal coefficient, or the ones determined by letting the nonzero diagonal elements be a doubling chain. We have not yet seen, however, whether these are all facets or not.

Theorem 3.5.7 (The facets of $\mathcal{C}_3$)**.** *The facet-defining inequalities of $\mathcal{C}_3$ are all the face-defining linear inequalities for which the nonzero diagonal coefficients (if any) form a doubling chain.*

Proof. From our discussion and Lemmas 3.5.5 and 3.5.6 we know that all the facets are found among these inequalities. We only need to prove that they do not imply each other, the rest follows from Theorem 1.7.1.

Assume two face-defining inequalities of $\mathcal{C}_3$ do not have the same zero diagonal coefficients. Then in one of them, call it F_1, there is some $a_{ii} = 0$ and such that the corresponding coefficient in the other, say F_2, is not zero. Then F_1 contains $\mathbf{e}_i\mathbf{e}_i^T$, and F_2 does not, thus satisfying 1.7.3. We only need to show that we can find such an element also if two of our inequalities have the same zero diagonal coefficients.

Assume F_1 and F_2 are two faces with at least two nonzero diagonal coefficients and with the same zeroes on the coefficient diagonal. Then, since they are not the same inequality, their nonzero diagonal coefficients are not the same, nor are they scalar multiples of one another. Since this is the case there exists some (i, j) such that in the inequality defining F_1, $-a_{ii} = -2a_{jj}$, while in the inequality defining F_2 this is not true. That means that F_1 contains the generator $(4\mathbf{e}_i + \mathbf{e}_j)(4\mathbf{e}_i + \mathbf{e}_j)^T$ while F_2 does not, and so we fulfill the criterion in Theorem 1.7.3.

This concludes our proof. □

Since $\mathcal{C}_3$ is full-dimensional we have not only found all the facet-defining inequalities, but we also know by Theorem 1.7.2 that they make up the unique minimal system defining $\mathcal{C}_3$ – and we see that PORTA did get it right when we applied it to obtain examples back in Section 3.5.1. It may be interesting to note that we can always describe the cone $\mathcal{C}_3$ in terms of inequalities with integer coefficients as a consequence of this theorem and Corollary 3.5.4.

Note also that while we would like to provide an example of a matrix in $\mathcal{C}_3$ which is completely positive that we could not know was completely positive before now, we would have to make it be of order at least 5, and verify an absurd amount of inequalities in order to show that it was in $\mathcal{C}_3$, so we have opted not to do this.

3.6 Conclusion

We considered two convex cones contained within $\mathcal{CP}_n$. One, the triple diagonal cone, we did not manage to handle, but for the other, $\mathcal{C}_3$, we determined a fairly nice description of the defining minimal system. That the test for inclusion in $\mathcal{C}_3$ and by extension $\mathcal{CP}_n$ this system provides is comparatively efficient seems unlikely – the number of required inequalities grows very fast. There are certainly at least 2^n doubling chains of a given length, and any distinct ordering gives rise to a unique facet-defining inequality.

It seems like the argument could be extended to cones $\mathcal{C}_k$ with $k > 3$ too, but as we already have so many inequalities we have not attempted this here – the cones are not likely to get nicer to work with.

Chapter 4

Linear Programming Tests

Thus far we have largely considered the completely positive cone from a theoretical perspective. In Chapter 2 we considered several of the more important results in the field, while in Chapter 3 we attempted to exploit the fact that $\mathcal{CP}_n$ is a cone to obtain a new test for complete positivity – in this we had some success, but the fact remains that in Lemma 2.1.10 we saw that if we want to obtain the entire cone of completely positive matrices we have to use an infinite set of generators. Considering the trouble we had already with a somewhat large finite set this does not seem like it will be a useful way of thinking if we are looking for a more general approach.

In this chapter, then, we will turn to numerical algorithms and attempt to find some that explicitly create a completely positive decomposition of some given matrix. As no explicit descriptions of the cone $\mathcal{CP}_n$ are known, the exact approach is out. We must instead turn to approximation. The general problem of determining complete positivity is of course a quadratic one, but linear programming has often turned out to be very useful even in problems that are not in themselves linear, and in this chapter we will, inspired by [15], see if we can use linear programming to approach the matter.

4.1 A general algorithm

The basic idea in this chapter is this: Assume we are given a matrix $A \in \mathcal{DNN}_n$. Then,

1. Pick a matrix $X_0 \in \mathbb{R}^{n\times k}$ for some k, preferrably so that A is at least somewhat close to $X_0X_0^T$.

2. Using some algorithm $\Omega : \mathbb{R}^{n\times k} \to \mathbb{R}^{n\times k}$ for choosing, we let $X_1 = \Omega(X_0)$, and we construct Ω in such a way that $X_1X_1^T$ is closer to A than $X_0X_0^T$ was.

3. We repeat step 2, obtaining a sequence $X_0, X_1, X_2, \ldots$ until we have reached some maximum number of iterations or $X_{t+1} \approx X_t$ (so we are reaching a stationary point of some sort).

4. If our final matrix X_N is such that $A \approx X_NX_N^T$ we conclude that $A \in \mathcal{CP}_n$. Otherwise, the test hasn't succeeded.

Note that since we are attempting to exhibit the completely positive factorisation explicitly, there is no danger of false positives (excepting round-off error issues, which we do not intend to go into in this thesis).

There are several vague points in the above outline. The two most important ones for us are how X_0 is chosen, and what we pick Ω to be. Of course, if we have no algorithm we have nothing, so our first order of business will be to describe two algorithms. Both will work the same way: We alternate between solving two linear programming problems and hope we get somewhere. The difference will be in their objective functions.

4.2 Alternating approximations

As mentioned, we cannot write the problem of complete positivity as a linear programming problem in an exact manner. However, assume that instead of trying to find an X with $A = XX^T$ we try to find $B \in \mathbb{R}_+^{n\times k}$, $C \in \mathbb{R}_+^{k\times n}$ such that the matrices $A - BC$, $B - C^T$ are in some sense small. If we let B_0 be a given matrix and solve said problem for C, call the computed matrix C_0, then solve the problem for B again, calling the solution B_1, and repeating, we will obtain a sequence of matrices $B_0, B_1, \ldots$ that hopefully exhibit some form of convergence. We make the procedure explicit.

Assume $f(A, B, C)$ where $A \in \mathcal{S}_+^n$, $B \in \mathbb{R}_+^{n\times k}$, $C \in \mathbb{R}_+^{k\times n}$ is a function taking values in $\mathbb{R}_+$ and the property that

$$f(A, B, C) = 0 \Leftrightarrow A = BC \text{ and } B = C^T. \tag{4.1}$$

Assume we are given some matrix A and an initial value B_0. For $k = 0, 1, \ldots$ define C_k by

$$C_k = \arg\min_{C\in\mathbb{R}_+^{k\times n}} f(A, B_k, C),$$

and for $k = 1, \ldots,$ define B_k by

$$B_k = \arg\min_{B\in\mathbb{R}_+^{n\times k}} f(A, B, C_{k-1}).$$

Above we take arg min to mean the value for which the minimum is attained. Obviously any solution of the above with $f(A, B, C) = 0$ will be a completely positive decomposition, which is what we are resting on throughout.

Before we can try implementing this, we need to come up with some choices of f. We are going to investigate two at first.

4.2.1 f_{max}

Consider the following linear problem, where the values m, n, a_{ij} and b_{ij} are assumed to be given:

$$\begin{array}{lrll}
\text{minimise} & r & & \\
\text{subject to} & r & \geq t_{ij} & \text{for } i = 1, \ldots, n, j = 1, \ldots, m \\
& r & \geq s_{ij} & \text{for } i = 1, \ldots, n, j = 1, \ldots, n \\
& t_{ij} & \geq b_{ij} - c_{ji} & \text{for } i = 1, \ldots, n, j = 1, \ldots, m \\
& t_{ij} & \geq c_{ji} - b_{ij} & \text{for } i = 1, \ldots, n, j = 1, \ldots, m \\
& s_{ij} & \geq a_{ij} - \sum_{k=1}^m b_{ik}c_{kj} & \text{for } i = 1, \ldots, n, j = 1, \ldots, n \\
& s_{ij} & \geq \sum_{k=1}^m b_{ik}c_{kj} - a_{ij} & \text{for } i = 1, \ldots, n, j = 1, \ldots, n \\
& c_{ij} & \geq 0 & \text{for } i = 1, \ldots, m, j = 1, \ldots, n
\end{array} \tag{MC}$$

Keeping in mind that $\|\cdot\|_{max}$ is the maximum element of a matrix (in absolute value), we see that (MC) is a linearisation[1] of the following problem:

$$\text{minimise} \max_{C \in \mathbb{R}_+^{m \times n}} \{\|B - C^T\|_{max}, \|A - BC\|_{max}\}. \tag{4.2}$$

Similarly, we could assume a_{ij} and c_{ij} are given, and consider the following problem:

$$\begin{array}{lllll}
\text{minimise} & r & & & \\
\text{subject to} & r & \geq t_{ij} & \text{for } i = 1, \ldots, n, j = 1, \ldots, m & \\
& r & \geq s_{ij} & \text{for } i = 1, \ldots, n, j = 1, \ldots, n & \\
& t_{ij} & \geq b_{ij} - c_{ji} & \text{for } i = 1, \ldots, n, j = 1, \ldots, m & \\
& t_{ij} & \geq c_{ji} - b_{ij} & \text{for } i = 1, \ldots, n, j = 1, \ldots, m & \\
& s_{ij} & \geq a_{ij} - \sum_{k=1}^{m} b_{ik} c_{kj} & \text{for } i = 1, \ldots, n, j = 1, \ldots, n & \\
& s_{ij} & \geq \sum_{k=1}^{m} b_{ik} c_{kj} - a_{ij} & \text{for } i = 1, \ldots, n, j = 1, \ldots, n & \\
& b_{ij} & \geq 0 & \text{for } i = 1, \ldots, n, j = 1, \ldots, m &
\end{array} \tag{MB}$$

This is a linearisation of the following problem:

$$\text{minimise} \max_{B \in \mathbb{R}_+^{n \times m}} \{\|B - C^T\|_{max}, \|A - BC\|_{max}\}. \tag{4.3}$$

Now we can define f_{max} for any given n, m:

$$\begin{aligned} f_{max}(A, B, C) = \max\{&\|B - C^T\|_{max}, \|A - BC\|_{max}\}, \\ &\text{where } A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}_+^{n \times m}, C \in \mathbb{R}_+^{m \times n}. \end{aligned} \tag{4.4}$$

Note that, being the maximum of two norms, f_{max} takes values in $\mathbb{R}_+$. We also need the following property:

Lemma 4.2.1. *f_{max} satisfies* (4.1).

Proof. If $A = BC$ and $B = C^T$ it follows that $\|A - BC\|_{max} = \|O_n\|_{max} = 0$ and $\|B - C^T\| = \|O_{n,m}\|_{max} = 0$, and so

$$f(A, B, C) = \max\{0, 0\} = 0.$$

If, on the other hand, $f_{max}(A, B, C) = 0$, then

$$\max\{\|B - C^T\|_{max}, \|A - BC\|_{max}\} = 0,$$

and since the two norms are nonnegative, being norms, it follows that both of them must be zero, and from the properties of norms it follows that $B - C^T = O_{n,m}$, $A - BC = O_n$, and we are done. □

We see that f_{max} as defined satisfies the conditions needed to be used as the basis for alternating approximations. Further, the two arg min-problems we need to solve are modelled linearly above, as (MB) and (MC).

[1] An equivalent problem which is linear.

4.2.2 f_{L1}

The function in this section is similar to the last one, but not entirely alike. Consider the following linear problem, where the values m, n, a_{ij} and c_{ij} are assumed to be given.

$$\begin{array}{lll} \text{minimize} & \sum_{i,j=1}^{n,m} t_{ij} + \sum_{i,j=1}^{n,n} s_{ij} & \\ \text{subject to} & t_{ij} \geq b_{ij} - c_{ji} & \text{for } i = 1, \ldots, n, j = 1, \ldots, m \\ & t_{ij} \geq c_{ji} - b_{ij} & \text{for } i = 1, \ldots, n, j = 1, \ldots, m \\ & s_{ij} \geq a_{ij} - \sum_{k=1}^{m} b_{ik} c_{kj} & \text{for } i = 1, \ldots, n, j = 1, \ldots, n \\ & s_{ij} \geq \sum_{k=1}^{m} b_{ik} c_{kj} - a_{ij} & \text{for } i = 1, \ldots, n, j = 1, \ldots, n \\ & c_{ij} \geq 0 & \text{for } i = 1, \ldots, m, j = 1, \ldots, n \end{array} \tag{LC}$$

Keeping in mind that the norm $\|\cdot\|_1$ on the matrix space is the sum of absolute values of elements in the given matrix, we see that (LC) is a linearisation of the following problem:

$$\text{minimise } \|B - C^T\|_1 + \|A - BC\|_1, C \in \mathbb{R}_+^{m \times n}.$$

Again we explicitly show the other problem too, assuming m, n, a_{ij} and b_{ij} are given:

$$\begin{array}{lll} \text{minimize} & \sum_{i,j=1}^{n,m} t_{ij} + \sum_{i,j=1}^{n,n} s_{ij} & \\ \text{subject to} & t_{ij} \geq b_{ij} - c_{ji} & \text{for } i = 1, \ldots, n, j = 1, \ldots, m \\ & t_{ij} \geq c_{ji} - b_{ij} & \text{for } i = 1, \ldots, n, j = 1, \ldots, m \\ & s_{ij} \geq a_{ij} - \sum_{k=1}^{m} b_{ik} c_{kj} & \text{for } i = 1, \ldots, n, j = 1, \ldots, n \\ & s_{ij} \geq \sum_{k=1}^{m} b_{ik} c_{kj} - a_{ij} & \text{for } i = 1, \ldots, n, j = 1, \ldots, n \\ & b_{ij} \geq 0 & \text{for } i = 1, \ldots, m, j = 1, \ldots, n \end{array} \tag{LB}$$

This is a linearisation of the following problem:

$$\text{minimise } \|B - C^T\|_1 + \|A - BC\|_1, B \in \mathbb{R}_+^{n \times m}.$$

The definition of f_{L1} is likely obvious now:

$$\begin{gathered} f_{L1}(A, B, C) = \|B - C^T\|_1 + \|A - BC\|_1, \\ \text{where } A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}_+^{n \times m}, C \in \mathbb{R}_+^{m \times n}. \end{gathered} \tag{4.5}$$

Being a nonnegative sum of norms, f_{l1} is a function taking values in $\mathbb{R}_+$. We again prove that it has the required property.

Lemma 4.2.2. *f_{L1} satisfies (4.1).*

Proof. If $A = BC$ and $B = C^T$ it follows that $\|A - BC\|_1 = \|O_n\|_1 = 0$ and $\|B - C^T\| = \|O_{n,m}\|_1 = 0$, and so

$$f_{L1}(A, B, C) = \max\{0, 0\} = 0.$$

If, on the other hand, $f_{L1}(A, B, C) = 0$, then

$$\|B - C^T\|_1 + \|A - BC\|_1 = 0,$$

and since the two norms are nonnegative, being norms, it follows that both of them must be zero, and from the properties of norms it follows that $B - C^T = O_{n,m}$, $A - BC = O_n$, and we are done. □

4.3 The little details

As we mentioned there are some matters beyond the choice of algorithm that require attention before we can try to implement them. We will not prove convergence of the algorithms, for reasons that will become obvious.

4.3.1 Initial guesses

Earlier we said we would need some initial guess X_0 (or B_0, as our notation in the previous section would have us call it). We shall see that the choice certainly does seem to matter. In general, we will let $k = \frac{n(n+1)}{2} - 1$ (see Section 2.4) so that we are not in danger of using too few columns in X_0 (In fact, we will almost certainly be using far too many, but the results concerning this are not conclusively proven). We will attempt three different choice strategies, all of which are simple.

Ones guess We can let X_0 be a matrix consisting of all ones.

Diagonal guess Define X_0 so that all elements on the diagonal are correct in $X_0 X_0^T$, i. e. $a_{ii} = (X_0 X_0)^T_{ii}$ for $i = 1, \ldots, n$.

Random guess We will also consider letting the initial matrix be entirely random – this will serve as a sort of check on our other strategies, by helping us determine whether they are helpful at all or not.

Of course, other methods of guessing may be available, but the ones here are at least different enough that they should let us see the impact of the initial value on the algorithms (if there is such an impact). The obvious way of defining the diagonal guess is the one we have used. Let A be given, and set

$$x_{ij} = \sqrt{\frac{a_{ii}}{m}} \text{ for } i = 1, \ldots, n, j = 1, \ldots, m.$$

Now we see that

$$(XX)^T_{ii} = \sum_{k=1}^{m} x_{ik} x_{ik} = \sum_{k=1}^{m} \frac{a_{ii}}{m} = a_{ii}.$$

This is what we have used throughout the present chapter as our diagonal guess.

4.3.2 Generating test cases and measuring error

We have described our algorithms, shown that they will eventually converge, and determined our initial values. Only a few things remain before we can investigate the algorithms in practice. First, we must discuss how we will determine that an algorithm has worked. Keep in mind that even if the algorithm finds B, C such that $A = BC$ it may not be the case that $BB^T = A$, and in general we might expect that BC is closer to A than BB^T is. We obviously need to measure the error of BB^T as an approximation to A, and we have chosen to use relative Frobenius error, which seems to be a widely used measure. The relative error we will use is defined as such:

$$\epsilon(A, B) = \frac{\|A - BB^T\|_F}{\|A\|_F}.$$

When later in the chapter we mention measured error, this is what we mean. As A, B are often clear from context we will usually refer to this simply as ϵ.

Note also that as we are interested in actual completely positive decompositions and not simply good approximations, we need ϵ to converge to 0. If it does not, the algorithm is useless to us, and if it does not approach 0 fast enough the algorithm is not very useful either. We will test every algorithm in this chapter for a maximum of 100 iterations, primarily because we only have access to an average computer, and even at an order of just 4 computing 100 iterations (which means solving 200 LP problems) takes a while. Also, if the algorithm appears to be stopping completely before then, we break off early. We define this as having the maximal elementwise difference in absolute value of the matrix $B_{k+1} - B_k$ be smaller than 10^{-10}. This is a very harsh requirement, but these algorithms sometimes take very small steps in between larger ones, and so we wish to be very sure we are not breaking off too early.

With error measurement sorted out, only one issue remains: What shall we test the algorithms on? Since they are all constructive, we need not fear false positives, so we only need to investigate whether they at least sometimes verify complete positivity of some matrix that we know is completely positive.

Precisely what kind of completely positive matrices are the "typical" ones in the cone $\mathcal{CP}_n$ is a rather difficult question to answer, so we shall instead work with a simple answer that is only somewhat satisfactory. We create a completely positive matrix by letting B be a random nonnegative matrix of the right dimensions, then setting $A = BB^T$. Then we "forget" B and make our guesses independently. In this way, we know that all the matrices we are testing are completely positive, but we can not know that this will give us a uniformly random distribution of matrices in $\mathcal{CP}_n$. With no other options readily available, however, this is what we will use.

For the sake of examples we will pick B to be a random nonnegative integer matrix with integers in the range 0-9. This may seem like it influences the algorithms too much, but some experiments suggest that the algorithm performs roughly the same by letting B simply be random, so we will stick to this picking strategy. We will "cheat" a little when making our random guess by letting the random numbers be in the same range, with the aim that if the random guess turns out to work well we will invent some other method which does not presuppose knowledge of the solution.

4.4 Results of alternating approximation

At this point we have made clear the way in which we intend to implement the algorithms we intend to test. They have been implemented in Java using IBM Ilog Concert Technology to call upon functions that solve linear programs quickly. The actual implementation is straightforward, and we shall not go into the details. Instead we move straight on to considering the results we obtain. We note that we have not always been able to test the algorithms on as many cases as we'd like, but we have at every step provided as many tested matrices as possible, making at least a few thousand per algorithm.

Table 4.1: Max-approximation data

Guess type	n	Avg. ϵ	Min. ϵ	Max. ϵ	Avg. iters	Test cases
Ones	4	9.19	7.28e-5	171.62	60	1000
	6	38.73	2.20e-4	825.30	87	500
	10	174.29	0.01	3322.72	100	250
	15	2791.85	0.32	25416.12	100	50
Diagonal	4	11.12	1.13	154.14	46	1000
	6	55.64	9.44e-4	784.53	81	500
	10	156.26	0.01	4194.14	99	250
	15	2217.26	22.65	14800.03	100	50
Random	4	0.34	4.28	74.69	42	1000
	6	0.03	1.32e-4	0.86	97	500
	10	0.04	0.003	0.09	100	250
	15	0.07	0.01	0.15	100	50

4.4.1 Results using f_{max}

First, we investigate what happens when using f_{max} as previously detailed – we will refer to this as *max-approximation*. The results we have gathered are summarised in Table 4.1. Before we start discussing the results, note that as the average number of iterations increases the number becomes less reliable since we always stop after 100.

We see that none of the three initial guess strategies seem to work all that well for max-approximation (In fact, they are all quite bad except for a rare few cases in which they seem to get lucky). Observe that for small n the algorithm is often capable of halting before 100 iterations, but as n increases we lose this property. We also observe that as expected the ones guess becomes worse as the dimension increases, but so does the diagonal guess, which we had hoped would stay closer.

Out of the three, the random guess by far performs best, but while the errors it obtains are not large, they are not yet small – the algorithm could conceivably perform better if we had the computing power to perform more iterations, but in truth running 100 iterations of the algorithm for a matrix of order 15 already takes several minutes. Thus we can hardly expect that the algorithm performs well in general, and we note that it often breaks off early at a solution with a somewhat large error, indicating that even if we could test it further it does not seem like it will commonly get there at all, rather stabilising some way off.

Before moving on we will consider a particular example, which clearly illustrates the weakness of our algorithm. In this case $n = 4$ and the algorithm terminated after 20 iterations, with an objective value of roughly 31, and a measured error of about 0.92. The matrix A was

$$A = \begin{bmatrix} 90 & 50 & 70 & 43 \\ 50 & 128 & 74 & 28 \\ 70 & 74 & 268 & 127 \\ 43 & 28 & 127 & 143 \end{bmatrix},$$

but the computed product BB^T after we stopped was

$$BB^T = \begin{bmatrix} 2.47 & 0.50 & 0.69 & 0.07 \\ 0.50 & 4.65 & 0.69 & 0.00 \\ 0.69 & 0.69 & 30.31 & 13.20 \\ 0.07 & 0.00 & 13.20 & 15.43 \end{bmatrix},$$

and this is just awful. To compare directly, the original matrix $B_{original}$ was

$$B_{original} = \begin{bmatrix} 7 & 2 & 4 & 1 & 1 & 3 & 1 & 3 & 0 \\ 0 & 9 & 4 & 2 & 1 & 4 & 1 & 0 & 3 \\ 0 & 0 & 3 & 0 & 9 & 7 & 4 & 8 & 7 \\ 0 & 0 & 0 & 7 & 5 & 2 & 1 & 8 & 0 \end{bmatrix},$$

while its computed replacement in this case was

$$B_{20} = \begin{bmatrix} 0.90 & 0.58 & 0.00 & 0.58 & 0.58 & 0.58 & 0.00 & 0.58 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.29 & 0.00 & 2.06 & 0.58 & 0.00 \\ 0.00 & 0.00 & 3.36 & 0.00 & 0.00 & 0.00 & 0.00 & 1.19 & 4.19 \\ 0.00 & 0.13 & 3.93 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \end{bmatrix}.$$

However, the computed product BC the algorithm obtained was

$$BC = \begin{bmatrix} 121.43 & 36.78 & 46.39 & 11.57 \\ 27.75 & 96.57 & 42.57 & 0.00 \\ 44.38 & 42.57 & 236.57 & 95.57 \\ 11.57 & 0.00 & 101.78 & 111.57 \end{bmatrix}.$$

We see that this is much better (while, unfortunately, still rather bad), and herein lies the trouble with max-approximation: The difference between B and C doesn't have to be very large in order for BB^T to be very different from BC, which is what keeps tripping up this particular algorithm. Also, as we are only measuring the maximal error out of many matrix elements, the algorithm changes its guesses little per step, as there is little incentive to do more. Therefore, we easily get stuck on local minima that are far from the best solution, and we conclude that the max-approximation algorithm does not do the job.

4.4.2 Results using f_{L1}

We refer to this kind of approximation as *L1-approximation*. Our results are summarised in Table 4.2.

Here too the random guess performs best, and we also observe that the algorithm appears to converge much faster. However, it often seems to converge to a point that is far from optimal. The main issue is that the weighting is somewhat off – the algorithm will decrease its objective function quickly by reducing the difference $A - BC$, while $B - C^T$ counts for much less.

We consider an example of using the diagonal guess before we move on. In this case $n = 4$, the objective function upon halting was roughly 67.7, and the error was about 0.67. The matrices involved are shown below.

$$A = \begin{bmatrix} 228 & 116 & 251 & 185 \\ 116 & 272 & 243 & 252 \\ 251 & 243 & 401 & 308 \\ 185 & 252 & 308 & 347 \end{bmatrix}.$$

Table 4.2: L1-approximation data

Guess type	n	Avg. ϵ	Min. ϵ	Max. ϵ	Avg. iters	Test cases
Ones	4	1.00	0.49	7.50	4	1000
	6	1.07	0.73	7.37	7	500
	10	1.21	0.68	10.99	12	250
	15	0.98	0.88	2.65	9	100
Diagonal	4	0.47	0.04	19.39	15	1000
	6	1.12	0.65	12.18	8	500
	10	0.24	0.06	0.70	74	250
	15	0.12	0.06	0.29	98	100
Random	4	0.38	0.02	5.14	9	1000
	6	0.20	0.03	0.55	36	500
	10	0.08	0.01	0.25	84	250
	15	0.04	0.8e-2	0.08	99	100

$$B = \begin{bmatrix} 6.12 & 0.00 & 2.32 & 2.56 & 12.99 & 2.06 & 0.00 & 14.84 & 5.31 \\ 0.14 & 2.98 & 5.49 & 8.64 & 6.20 & 21.12 & 0.00 & 0.00 & 0.00 \\ 5.49 & 6.61 & 5.98 & 12.84 & 11.69 & 4.86 & 0.00 & 15.53 & 3.75 \\ 7.84 & 8.65 & 6.21 & 12.82 & 10.74 & 5.45 & 2.68 & 0.00 & 2.78 \end{bmatrix}.$$

$$BB^T = \begin{bmatrix} 470.78 & 159.71 & 491.60 & 260.78 \\ 159.71 & 598.12 & 336.28 & 353.50 \\ 491.60 & 336.28 & 680.72 & 459.89 \\ 260.78 & 353.50 & 459.89 & 499.42 \end{bmatrix}.$$

$$C = \begin{bmatrix} 6.12 & 0.14 & 5.49 & 7.84 \\ 0.00 & 2.98 & 6.61 & 8.65 \\ 2.32 & 5.49 & 5.98 & 6.21 \\ 2.56 & 7.73 & 7.93 & 4.63 \\ 5.94 & 5.36 & 2.98 & 6.44 \\ 2.06 & 6.29 & 4.86 & 5.26 \\ 0.00 & 0.00 & 0.00 & 2.68 \\ 4.65 & 0.00 & 7.72 & 0.11 \\ 5.31 & 0.00 & 3.75 & 2.78 \end{bmatrix}.$$

$$BC = \begin{bmatrix} 228 & 116 & 251 & 185 \\ 116 & 272 & 243 & 252 \\ 251 & 243 & 401 & 308 \\ 185 & 252 & 308 & 347 \end{bmatrix}.$$

From the above we see that the algorithm actually performs a nonnegative matrix factorisation perfectly in this case, but after reaching this point there is comparatively so little to gain by reducing the difference $B - C^T$ as opposed to an eventual gain in the difference $A - BC$, and so the algorithm in this case, too, gets stuck at a non-optimal solution which is somewhat off from what we were hoping for. Before we give up entirely on linear programming-based approaches we will try to weight the terms a little against each other and see if we obtain better results.

Table 4.3: Weighted L1-approximation data

α	n	Avg. ϵ	Min. ϵ	Max. ϵ	Avg. iters	Test cases
0	4	0.87	0.75	0.92	2	1000
	6	0.94	0.93	0.95	2	500
	10	0.84	0.83	0.85	5	250
	15	0.94	0.94	0.94	10	100
0.5	4	0.21	0.05	0.47	3	1000
	6	0.09	0.3e-2	0.21	29	500
	10	0.11	0.04	0.23	57	250
	15	0.05	0.02	0.10	94	100
2	4	0.44	0.14	1.14	1	1000
	6	0.43	0.23	0.62	1	500
	10	0.42	0.35	0.49	1	250
	15	0.41	0.37	0.44	1	100

4.4.3 Results using $f_{\alpha,L1}$

We define $f_{\alpha,L1}$ by

$$f_{\alpha,L1}(A,B,C) = \alpha\|B - C^T\|_1 + \|A - BC\|_1, \\ \text{where } A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}_+^{n\times m}, C \in \mathbb{R}_+^{m\times n}, \alpha \in \mathbb{R}_+. \tag{4.6}$$

We intend to use this as the basis for an alternating approximation in a manner that is entirely analoguous to the way we used f_{L1}. All the rest regarding errors and convergence remains the same as it was. We test this algorithm using the diagonal guess, as the random guess we have been using is not entirely honest. We investigate a few different values of α, see Table 4.3.

The gist of what we are seeing is that the idea is not all bad – it does seem to provide slightly better approximations – but it is not so good that we quite get good solutions either. It is difficult to say what α ought to be if we want good results, and we will leave the trail here, hoping for better results in the next chapter.

4.5 Conclusion

We have investigated several algorithms based on linear programming, hoping to uncover a test for complete positivity. Sadly, none of them have been great successes, though they do exhibit some of the features one would expect. The best guessing strategy we have found is to let the matrix be a random matrix that is likely to get close, but even this does not appear to work very well with our algorithms.

Therefore we have not in this chapter gone into finding some way of making such a random matrix *without* foreknowledge of its approximate range, nor have we attempted to provide a proper proof of the algorithms' convergence – it seems like a waste of the reader's time given that they do not actually exhibit

the properties we need – but we do note that they all exhibited convergent behaviour in practice.

In the next chapter we shall implement an algorithm based on the descent method – thus we will remain on the mathematical optimisation track, but we are leaving linearity of the objective function behind.

Chapter 5

Descent Method

The approach of chapter 4 failed us. In general, we saw some success at solving NMF-type problems, but nothing like real success at approaching the problem of complete positivity. The main problem was that we could not get $B = C^T$ emphasised enough without losing emphasis on $A = BC$. In this chapter we will instead attempt to use a so-called *steepest descent method* to approach the issue, in which we will always be working with XX^T as an approximation to A, rather than the approach of having two matrices in the previous chapter. The algorithm and its convergence properties are taken from the excellent book [7].

5.1 The descent method algorithm

The algorithm we will be using is described as follows. Assume we are given a function $f_A(X)$ such that $f_A(X) = 0$ if and only if $A = XX^T$.

1. Choose a starting guess $X \in \mathbb{R}_+^{n \times m}$.
2. The descent direction is chosen to be $-\nabla f_A(X)$, which is the direction of steepest descent.
3. A step size $\mu > 0$ is chosen such that $f_A(X - \mu \nabla f_A(X)) < f_A(X)$.
4. We define the new starting point as $X - \mu \nabla f_A(X)$ and begin again from step 2. If some stopping criterion is satisfied we quit, if not we return to step 2 and continue from there.

It should be obvious what the above algorithm does. We pick a step direction which is known to be the one in which the function value decreases fastest, then obtain a step length such that we actually decrease the function value, then we repeat this process.

The above described algorithm is not complete, however, and in particular we need to make explicit what it means to pick a step size μ. In general, the descent direction defines a ray $\{X - t\nabla X : t \geq 0\}$, and we are looking for the minimizer of f along that ray. Solving the problem explicitly is often difficult, however, so instead a good heuristic is usually used. These issues are, as mentioned, more clearly described in [7], and from that book we take the idea of using *backtracking line search*. This is defined in the following manner.

Given a descent direction ΔX for f at some X, let $\alpha \in (0, 0.5), \beta \in (0, 1)$, define t to be equal to 1. For as long as

$$f(X + t\Delta x) > f(X) + \alpha t \nabla f(x)^T \Delta x$$

holds, redefine t by $t := \beta t$. This builds on a first-order Taylor approximation, and it is known to converge. We will use $\alpha = 0.1$ and $\beta = 0.3$ in our algorithm.

5.2 Steepest descent

Before we can attempt implementing the algorithm we must determine our function $f_A(X)$ and its properties. We assume n is some positive integer, $A \in \mathbb{R}_+^{n \times n}$ and m is another integer, in general equal to $\frac{n(n+1)}{2} - 1$.

So, we assume $A = [a_{ij}]_{i,j=1}^n$ and $X = [X_{ij}]_{i,j=1}^{n,m}$. We assume $X \geq 0$. Now we can define the following function(Recall the definition of the Frobenius norm in Section 1.4):

$$f_A(X) = \|A - XX^T\|_F^2.$$

Of course, if we can find an X with $f_A(X) = 0$ we have proved complete positivity of the matrix A, and for all completely positive matrixes A such an X must exist by definition, and so A is completely positive if and only if there exists an X such that $f_A(X) = 0$. For this reason we will study f_A.

First a note on computation, for algorithmic purposes. We see that

$$XX^T = \begin{bmatrix} x_{11} & \cdots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nm} \end{bmatrix} \begin{bmatrix} x_{11} & \cdots & x_{n1} \\ \vdots & \ddots & \vdots \\ x_{1m} & \cdots & x_{nm} \end{bmatrix},$$

from which it follows that the (i, j)-th element of XX^T is

$$(XX^T)_{ij} = \sum_{k=1}^{m} x_{ik} x_{jk}.$$

Using this we can write out f_A in terms of the matrix elements, obtaining the expression

$$f_A(X) = \sum_{i,j=1}^{n} \left(a_{ij} - \sum_{k=1}^{m} x_{ik} x_{jk} \right)^2. \tag{5.1}$$

If we want to compute f_A by way of vector operations, the following is a simple formulation:

$$f_A(X) = \mathbf{1}_n^T (A - XX^T) \circ (A - XX^T) \mathbf{1}_n.$$

In general, f_A is hard to visualise because there are very many variables. We would very much like to show how the function looks in at least one case, though, so let us consider a rank 1 completely positive matrix,

$$\begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} 1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}.$$

If we let $n = 2$, $m = 1$ we can look for $X \in \mathbb{R}^{2\times 1}$. In this case we can actually write out the function f_A, letting $X = \begin{bmatrix} x \\ y \end{bmatrix}$:

$$f_A(X) = \left(25 - 2x^2 - 8xy - 8y^2 + x^4 + 2x^2y^2 + y^4\right)^2.$$

We use this to create an illustration, letting $x, y \in [0, 10]$. See figure 5.1 for an illustration of how it looks. What we can take away from this is that the function does not immediately appear to be very "ugly", as it were. It instead appears to be rather nice and smooth, with the function taking the smallest values around the area in which the elements of X are such that their products are approximately right for A, and increasing quite rapidly when we leave that area. This is encouraging.

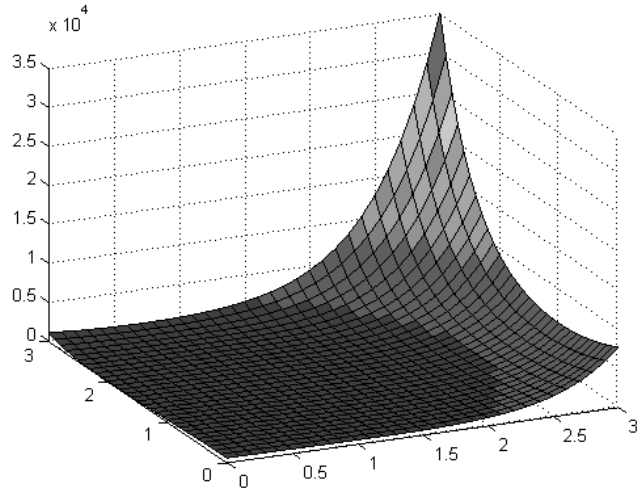


Figure 5.1: Illustration of f_A.

5.2.1 f_A is not a convex function

Convexity is a very important property for functions to have in any kind of minimization situation (we assume the reader is familiar with convex functions). Unfortunately, f_A is not a convex function, at least not in general. We give a short example to prove that this may not be true. Assume $n = 2$ and $m = 1$. Then X is a 2-by-1 matrix, and for the sake of example we can assume only the first element of A is nonzero. Further, assume the second element in X is zero, and call the first one x. This corresponds to inspecting f_A for convexity along a line (or ray, given our domain requirements).

So, if $x \geq 0$, the function is

$$\begin{aligned} f_A(x) &= (a_{11} - x^2)^2 + (a_{12} - 0)^2 + (a_{21} - 0)^2 + (a_{22} - 0)^2 \\ &= x^4 - 2a_{11}x^2 + a_{11}^2 + a_{12}^2 + a_{21}^2 + a_{22}^2. \end{aligned}$$

Technically, this is the restriction of $f_A(X)$ to the ray in the space $\mathbb{R}^{2\times 1}$ where $x_{11} \geq 0, x_{21} = 0$, but we understand what we mean by $f_A(x)$. Now we will show that this function is not always convex. It is easy enough to see. Recall that a function on $\mathbb{R}$ is only convex if its double derivative is always nonnegative. In the case above,

$$f_A''(x) = 12x^2 - a_{11},$$

and we could for instance assume $a_{11} = 12$, meaning that the function is negative for all $x \in [0, 1)$. From this it follows that $f_A(x)$ is not, in general, convex, and then it follows that $f_A(X)$ is not convex, and so we cannot assume in general that our objective function is convex – and in fact we shall see later that there are in general several solutions of $f_A(X) = 0$ in the nonnegative orthant.

5.2.2 The gradient of f_A

We want to compute the gradient of f_A, and for the sake of completeness we try to find a nice formula for it. f_A is a function on a matrix space, and we define the gradient of f_A as if it were a function on a vector space, "pretending" our matrices have been vectorised. We compute the gradient explicitly, element by element, then summarise our findings at the end of this section.

We begin by considering the component form of f_A (see (5.1)):

$$f_A(X) = \sum_{i,j=1}^{n} \left(a_{ij} - \sum_{k=1}^{m} x_{ik}x_{jk} \right)^2.$$

Now, let r, s be fixed integers with $1 \leq r \leq n$, $1 \leq s \leq m$. Then

$$\begin{aligned}
(\nabla f_A(X))_{rs} &= \frac{\partial}{\partial x_{rs}} f_A(X) \\
&= \frac{\partial}{\partial x_{rs}} \sum_{i,j=1}^{n} \left(a_{ij} - \sum_{k=1}^{m} x_{ik}x_{jk} \right)^2 \\
&= \sum_{i,j=1}^{n} \frac{\partial}{\partial x_{rs}} \left(a_{ij} - \sum_{k=1}^{m} x_{ik}x_{jk} \right)^2 \\
&= 2 \left(\sum_{i,j=1}^{n} \left(a_{ij} - \sum_{k=1}^{m} x_{ik}x_{jk} \right) \frac{\partial}{\partial x_{rs}} \left(a_{ij} - \sum_{k=1}^{m} x_{ik}x_{jk} \right) \right) \\
&= 2 \left(\sum_{i,j=1}^{n} \left(a_{ij} - \sum_{k=1}^{m} x_{ik}x_{jk} \right) \left(-\sum_{k=1}^{m} \frac{\partial}{\partial x_{rs}} x_{ik}x_{jk} \right) \right) \\
&= -2 \left(\sum_{i,j=1}^{n} \left(a_{ij} - \sum_{k=1}^{m} x_{ik}x_{jk} \right) \left(\sum_{k=1}^{m} \frac{\partial}{\partial x_{rs}} x_{ik}x_{jk} \right) \right).
\end{aligned}$$

Now consider the term $\frac{\partial}{\partial x_{rs}} x_{ik}x_{jk}$. If neither i nor j are equal to r, this term is zero. We split into the three cases $i = r, j \neq r$, $i \neq r, j = r$ and $i = j = r$. In any case the term is nonzero only when $k = s$, and we compute the nonzero

terms of the above sum as follows:

$$\begin{aligned} i = r, j \neq r: \quad & -2 \sum_{j=1, j\neq r}^{n} \left(\left(a_{rj} - \sum_{k=1}^{m} x_{rk} x_{jk} \right) \left(\sum_{k=1}^{m} \frac{\partial}{\partial x_{rs}} x_{rk} x_{jk} \right) \right) \\ & = -2 \sum_{j=1, j\neq r}^{n} \left(a_{rj} - \sum_{k=1}^{m} x_{rk} x_{jk})(x_{js}) \right) \\ & = -2 \sum_{j=1, j\neq r}^{n} x_{js} \left(a_{rj} - \sum_{k=1}^{m} x_{rk} x_{jk} \right), \end{aligned} \tag{5.2}$$

$$\begin{aligned} i \neq r, j = r: \quad & -2 \sum_{i=1, i\neq r}^{n} \left(\left(a_{ir} - \sum_{k=1}^{m} x_{ik} x_{rk} \right) \left(\sum_{k=1}^{m} \frac{\partial}{\partial x_{rs}} x_{ik} x_{rk} \right) \right) \\ & = -2 \sum_{i=1, i\neq r}^{n} \left(\left(a_{ir} - \sum_{k=1}^{m} x_{ik} x_{rk} \right) (x_{is}) \right) \\ & = -2 \sum_{i=1, i\neq r}^{n} x_{is} \left(a_{ir} - \sum_{k=1}^{m} x_{ik} x_{rk} \right), \end{aligned} \tag{5.3}$$

$$\begin{aligned} i = j = r: \quad & -2 \left(a_{rr} - \sum_{k=1}^{m} x_{rk} x_{rk} \right) \left(\sum_{k=1}^{m} \frac{\partial}{\partial x_{rs}} x_{rk} x_{rk} \right) \\ & = -4 \left(a_{rr} - \sum_{k=1}^{m} x_{rk} x_{rk} \right) (x_{rs}) \\ & = -2 x_{rs} \left(a_{rr} - \sum_{k=1}^{m} x_{rk} x_{rk} \right) - 2 x_{rs} \left(a_{rr} - \sum_{k=1}^{m} x_{rk} x_{rk} \right). \end{aligned} \tag{5.4}$$

The reason for splitting the last case (5.4) into two as we did above is to illustrate that it fits into the two previous sums in such a way that we can remove the stipulations $j \neq r$ in (5.2) and $i \neq r$ in (5.3). Then writing this element of the gradient simply as the sum of the expressions we then obtain:

$$(\nabla f_A(X))_{rs} = -2 \sum_{j=1}^{n} x_{js} \left(a_{rj} - \sum_{k=1}^{m} x_{rk} x_{jk} \right) - 2 \sum_{i=1}^{n} x_{is} \left(a_{ir} - \sum_{k=1}^{m} x_{ik} x_{rk} \right).$$

Note that since A is by assumption a symmetric matrix $a_{rt} = a_{tr}$ for all t. Now, the two sums above are over the same range, so we combine them into one sum, and we get the following expression, where we also interchanged the order of

factors in the product in the rightmost sum above:

$$\begin{aligned}
\left(\nabla f_A(X)\right)_{rs} &= -2\sum_{t=1}^{n}\left(x_{ts}\left(a_{rt}-\sum_{k=1}^{m}x_{rk}x_{tk}\right)+x_{ts}\left(a_{rt}-\sum_{k=1}^{m}x_{rk}x_{tk}\right)\right)\\
&= -4\sum_{t=1}^{n}x_{ts}a_{rt}+4\sum_{t=1}^{n}x_{ts}\sum_{k=1}^{m}x_{rk}x_{tk}\\
&= -4\sum_{t=1}^{n}a_{rt}x_{ts}+\sum_{t=1}^{n}x_{ts}\left(XX^T\right)_{rt}\\
&= -4(AX)_{rs}+4\sum_{t=1}^{n}\left(XX^T\right)_{rt}x_{ts}\\
&= -4(AX)_{rs}+4\left(\left(XX^T\right)X\right)_{rs}=4\left(\left(XX^T-A\right)X\right)_{rs},
\end{aligned}$$

and so we can conclude that

$$\nabla f_A(X)=4\left(XX^T-A\right)X.$$

5.2.3 Summary of f_A

We take a moment to sum up the properties of the function f_A. They were all explained above.

Proposition 5.2.1 (Properties of f_A)**.** *1. The function f_A can be computed either through matrix operations according to the formula*

$$f_A(X)=\mathbf{1}_n^T(A-XX^T)\circ(A-XX^T)\mathbf{1}_n,$$

or more directly in terms of the elements using the equivalent formula

$$f_A(X)=\sum_{i,j=1}^{n}\left(a_{ij}-\sum_{k=1}^{m}x_{ik}x_{jk}\right)^2.$$

2. Its gradient ∇f_A can be computed either directly with matrix operations as

$$\nabla f_A(X)=4(A-XX^T)X,$$

or by calculating each element (r,s) individually using the formula

$$\left(\nabla f_A(X)\right)_{rs}=4\left(\sum_{t=1}^{n}x_{ts}\left(\sum_{k=1}^{m}x_{rk}x_{tk}-a_{rt}\right)\right).$$

3. f_A is not a convex function.

5.3 Results using the steepest descent method

We have described the steepest descent algorithm as it is explained in [7]. We will not go into implementation details, as they are hopefully obvious from the previous exposition. However, we note that if we try to implement this directly

Table 5.1: Projection descent algorithm with random initial guess

Tolerance	n	Avg. ϵ	Min. ϵ	Max. ϵ	Avg. iters	Test cases
1e-08	4	4.73e-09	8.67e-10	2.28e-08	331	1000
	6	3.99e-09	1.75e-09	1.40e-08	347	1000
	10	6.96e-09	2.36e-09	1.20e-08	513	1000
	15	7.38e-09	3.58e-09	1.01e-08	650	500
	20	4.23e-09	2.92e-09	5.78e-09	596	500
1e-10	4	4.78e-11	8.05e-12	2.26e-10	441	1000
	6	4.03e-11	1.87e-11	1.07e-10	461	1000
	10	7.05e-11	2.35e-11	1.32e-10	690	1000
	15	7.36e-11	2.94e-11	1.03e-10	869	500
	20	4.23e-11	2.83e-11	5.81e-11	788	500
1e-12	4	8.73e-13	1.11e-13	3.97e-10	530	1000
	6	4.00e-13	1.51e-13	9.77e-13	577	1000
	10	6.87e-13	2.28e-13	1.13e-12	848	1000
	15	7.35e-13	3.04e-13	1.13e-12	1100	500
	20	4.20e-13	2.93e-13	5.98e-13	995	500

we will see that the results are unsatisfactory, for we have not yet discussed how to maintain the nonnegativity of X in every step. There are several possible approaches to this, but the one we have had the most success with is simply projecting it on the nonnegative orthant after each iteration, which is to say that we shall set the negative elements of X to be equal to 0. This is mentioned in [5] as a strategy that is often used in similar algorithms (in particular, algorithms for nonnegative matrix factorisation) to avoid going outside the nonnegative orthant.

We generate random matrices as in the previous chapter, though instead of random integers between 0 and 9 we are using random floating point numbers between 0 and 1000, to alleviate any uneasiness about our "testing space" being too small. We also measure error as in the previous chapter, and we end the iteration when the change in X_i per iteration becomes smaller than some given tolerance δ, according to the condition

$$\|X_i - X_{i-1}\|_{max} < \delta,$$

which seems as good as any other and has the advantage that it is easy to use.

Both our so-called ones choice and diagonal choice methods from last chapter are completely unusable in this situation. The reason for this lies with the gradient we found. In both the ones guess and the diagonal guess the columns are all identical, and our algorithm will preserve this property throughout all iterations. This is useless, as we obviously cannot expect the matrix A to be representable as a sum of linearly dependent rank 1 matrices. Therefore, out of the previously tried initial values, only the random guess remains as a possibility, and we investigate that first. We still "cheat" in the same way, though. See Table 5.1.

We observe that we use far more iterations with this method, but on the

other hand they are much quicker – an approximation with several thousand iterations here is still faster than one using 100 in the previous chapter. Second, the method seems to work, remarkably. By lowering our target tolerance we can even get better results than we are, but it seemed prudent to compare our algorithms as similarly as possible, and with a smaller tolerance we would be approaching the limits of machine precision, which we are not going to consider here. We conclude for the moment that this algorithm does seem like it may be useful in determining complete positivity.

One thing remains unsaid, however. We have cheated in our initial guess; we know that the random matrix is in roughly the right range. We wish to exhibit a different initial guess choice that does not depend on this. We do this by picking a random nonnegative matrix and scaling the rows such that they match on the diagonal, in a mixture of the diagonal and the random guess. Let $A \in \mathbb{R}_+^{n\times n}$ be given and let $X' \in \mathbb{R}_+^{n\times m}$ be some randomly chosen matrix, and let

$$\S_i = \sqrt{\sum_{j=1}^{n} x_{ij}^2},$$

that is, $\S_i$ is the norm of the i-th row in X'. Let the numbers on the diagonal be defined by letting

$$d_i = \frac{\sqrt{a_{ii}}}{\S_i}.$$

Now define $X = DX'$, and consider the product XX^T. The i-th diagonal element of this product is

$$(XX)_{ii}^T = (DX'X'^T D)_{ii} = \frac{a_{ii}}{\S_i^2}\S_i^2 = a_{ii}.$$

Thus, this scaling ensures that our initial guess is correct for the diagonal elements.

In Table 5.2 we see the results we obtain. We can certainly claim they appear to be as good as the unmodified random guess, and so we have obtained an algorithm that presupposes no particular knowledge of the solution to work yet is still very good. The short summary of our situation is that the projection gradient method is empirically a success. It gives a correct, positive answer for all of our randomly generated completely positive matrices, and it does so in a reasonable timeframe for small matrices (on the average computer this was tested on, the computation for the 500 matrices of order 20 took less than ten minutes).

One example is warranted. We pick a small one. In this example we used a tolerance of 10^{-12}, set $n = 4$ and concluded after 313 iterations. The computed error at this point was 1.99e-13, which is enough that we will call this successful convergence. The original B matrix was

$$B = \begin{bmatrix} 8 & 6 & 9 & 9 & 4 & 6 & 6 & 6 & 2 \\ 9 & 0 & 9 & 4 & 9 & 0 & 7 & 1 & 0 \\ 1 & 2 & 1 & 8 & 7 & 8 & 7 & 7 & 0 \\ 9 & 5 & 9 & 1 & 9 & 9 & 3 & 0 & 8 \end{bmatrix}.$$

Table 5.2: Projection descent algorithm with modified initial guess

Tolerance	n	Avg. ϵ	Min. ϵ	Max. ϵ	Avg. iters	Test cases
1e-08	4	4.45e-11	5.00e-12	1.69e-10	276	1000
	6	4.23e-11	6.11e-12	9.95e-11	254	1000
	10	4.16e-11	2.07e-11	6.68e-11	482	1000
	15	4.26e-11	2.55e-11	6.52e-11	571	1000
	20	4.51e-11	9.49e-12	7.61e-11	357	500
1e-10	4	1.09e-11	9.42e-14	1.05e-08	351	1000
	6	4.11e-13	5.03e-14	1.16e-12	318	1000
	10	4.10e-13	2.41e-13	7.81e-13	606	1000
	15	4.22e-13	2.92e-13	6.54e-13	718	500
	20	4.53e-13	3.85e-14	8.06e-13	447	500
1e-12	4	4.57e-15	6.04e-16	2.16e-14	434	1000
	6	4.07e-15	5.94e-16	1.08e-14	371	1000
	10	4.05e-15	2.33e-15	7.68e-15	724	1000
	15	4.21e-15	2.51e-15	6.55e-15	856	500
	20	4.51e-15	7.86e-16	7.92e-15	534	500

This gives us

$$A = \begin{bmatrix} 390 & 273 & 261 & 316 \\ 273 & 309 & 169 & 268 \\ 261 & 169 & 281 & 192 \\ 316 & 268 & 192 & 423 \end{bmatrix}.$$

The computed completely positive decomposition $A = XX^T$ had (after rounding)

$$X = \begin{bmatrix} 7.04 & 5.27 & 2.36 & 6.80 & 6.56 & 9.33 & 7.93 & 7.93 & 2.20 \\ 2.15 & 4.14 & 6.24 & 9.70 & 1.83 & 2.99 & 2.51 & 11.45 & 2.16 \\ 9.87 & 6.26 & 2.70 & 0.85 & 0.71 & 5.97 & 4.30 & 4.35 & 7.93 \\ 0.18 & 10.06 & 8.33 & 9.60 & 7.01 & 3.30 & 9.32 & 2.55 & 2.63 \end{bmatrix}.$$

The difference $A - XX^T$ is very small, so we do not include it here. We do note that we have not found the completely positive decomposition we started with, but that is no surprise, for as we have said earlier we can not expect that there is only one.

It may be of interest to consider a matrix which is known not to be completely positive as well. We do not have many results which give us non-completely positive yet doubly nonnegative matrices, but in the discussing following Corollary 2.3.2 we showed that the matrix

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 1 & 0 \\ 0 & 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 2 & 4 \end{bmatrix}$$

is not completely positive. With a tolerance of 10^{-12} our algorithm stabilises at a matrix X such that

$$XX^T = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3.02 & 0.94 & 0.03 \\ 0 & 0 & 0.94 & 1.19 & 1.91 \\ 0 & 0 & 0.03 & 1.91 & 4.04 \end{bmatrix}.$$

The relative Frobenius error in this case is 0.038. It does seem like we managed to get somewhat close, but obviously we did not quite get there (and it would be a great surprise if we did!).

5.4 Conclusion

At this point we would very much like to offer the reader some proof of convergence, or failing that, an indication as to which corner cases the algorithm does not work for. Unfortunately, however, we have not been able to find a completely positive matrix the algorithm does not work for, nor have we been able to prove that it will work in general[1].

It does seem like the algorithm we have exhibited will in general be somewhat efficient. Recalling 1.10 we note that matrix multiplication is known to be polynomial-time, and it is easy enough to compute a bound on the backtracking line search (sooner or later, we will suffer underflow in an implementation), so the main issue is determining the number of iterations we need. That we have not been able to do – it would be tantamount to providing a convergence analysis, and as mentioned we have not been able to do this.

Therefore we must be careful about what we say, for we can not know that this works in general. What we do know is that if the algorithm in this chapter stops with a Frobenius error that is small in some sense (depending on what we are doing with the matrices) we feel that we can say the matrix is completely positive. The algorithm will not provide false positives. Further, we can state that we have seen that the algorithm works well for a few thousand matrices[2], though their randomness in the space $\mathcal{DD}_n$ is not entirely clear. Nevertheless, we have created a test that at least works in some cases.

[1] Given the difficulties involved in determining complete positivity of a matrix this seems like it is a little much to hope for.

[2] Of which only a few were shown to be completely positive by the sufficiency-type results in Chapters 2 and 3.

Chapter 6

Conclusion

In the introduction we stated that we were going to do three things. We have done all three. After covering the main points of the theory in Chapter 2, we moved on in Chapter 3 to considering various ways of enlarging the cone of nonnegative diagonally dominant matrices. In particular, we tried two approaches, one of which did not work, and one which did, thus obtaining a class of completely positive matrices we could determine membership of in an explicit manner.

In the final two chapters, we investigated various algorithms for providing inexact tests of complete positivity. The alternating minimisation approach of Chapter 4 did not provide us with much we can use, but the projection descent method of Chapter 5 did indeed work very well for the cases we tested.

Some interesting questions suggested by the work we have done are:

- Are there other extensions of $\mathcal{DD}_n$ which could work? We proved that $\mathcal{C}_3$ worked fine, and the proof seems like it could generalise somewhat and work for $\mathcal{C}_4$ and perhaps even higher, though how much use it will be is perhaps dubious.
- The projection descent method seems to work, but many small questions remain here:
  - How well does it work? We provided only empirical evidence that it was a good algorithm in some cases.
  - For which completely positive matrices does it work?
  - For which completely positive matrices does it not work?
  - Will it even always converge at all? If not, can it be modified so it does?
  - What is a good heuristic for choosing a starting point? Our modified random guess works reasonably well, but better heuristics may be possible.

Bibliography

[1] F. Barioli and A. Berman, *The maximal cp-rank of rank k completely positive matrices*, Linear Algebra and its Applications **363** (2003), 17–33.

[2] A. Berman and U. G. Rothblum, *A note on the computation of the cp-rank*, Linear Algebra and its Applications **419** (2006), 1–7.

[3] A. Berman and N. Shaked-Monderer, *Completely positive matrices*, World Scientific Publishing Co. Pte. Ltd., 2003.

[4] A. Berman and C. Xu, *{0,1} completely positive matrices*, Linear Algebra and its Applications **399** (2005), 35–51.

[5] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons, *Algorithms and applications for approximate nonnegative matrix factorization*, Computational Statistics and Data Analysis **52** (2007), no. 1, 155–173.

[6] I. M. Bomze, M. Dür, E. De Klerk, C. Roos, A. J. Quist, and T. Terlaky, *On copositive programming and standard quadratic optimization problems*, Journal of Global Optimization **18** (2000), 301–320.

[7] S. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge University Press, 2004.

[8] G. Dahl, *A note on diagonally dominant matrices*, Linear algebra and its applications **317** (2000), 217–224.

[9] ———, *An introduction to convexity*, 2009.

[10] C. L. Hamilton-Jester and C.-K. Li, *Extreme vectors of doubly nonnegative matrices*, The Rocky Mountain Journal of Mathematics **26** (1993), no. 4, 1371–1383.

[11] D. C. Lay, *Linear algebra and its applications*, Pearson Education, Inc., 2006.

[12] Y. Li, A. Kummert, and A. Frommer, *A linear programming based analysis of the cp-rank of completely positive matrices*, Int. J. Appl. Math. Comput. Sci. **14** (2004), no. 1, 25–31.

[13] T. Lyche, *Lecture notes for inf-mat 4350, 2010*, 2010.

[14] A. Schrijver, *Theory of linear and integer programming*, Wiley, 1998.

[15] J. A. Tropp, *An alternating minimization algorithm for non-negative matrix approximation*,
http://www.acm.caltech.edu/~jtropp/pubs.html, 2003.

[16] R. J. Vanderbei, *Linear programming*, Springer Science+Business Media, 2008.

[17] M. A. Weiss, *Data structures and algorithm analysis in java*, Pearson Education, Inc., 2007.

[18] C. Xu, *Completely positive matrices*, Linear Algebra and its Applications **379** (2004), 319–327.