Linear and Semidefinite Programming

K Prahlad Narasimhan*

National Institute of Science Education and Research, HBNI, Bhubaneswar, India

January 2021

In this report, we will discuss two linear optimiization constructs: linear programming and semidefinite programming. The goal of these two constructs is to optimize a linear function subject to several constraints. In linear programming, we require the constraints to be linear while in semidefinite programming we also impose that the variables describe a positive matrix. We will use linear and semidefinite programming to design approximation algorithms for computationally hard problems.

In Section 1, we define a linear program and use it to design algorithms to solve the Set Cover problem (in Subsection 1.1) and the Max-Flow problem (in Subsection 1.2). We prove a fundamental result in linear programming: the *Strong Duality Theorem* using *Farkas' Lemmas* in Section 2. We will then move onto characterizing positive matrices in Section 3 and and use that result in Section 4 to define semidefinite programming. Finally, we will present an algorithm using semidefinite programming to solve the Max-Cut problem.

1 Linear Programming

In *linear programming*, we are required to find a vector x (or report that no such vector exists) that optimizes (maximizes or minimizes) a given linear *objective function* defined on x subject to linear *constraints* (inequalities or equalities) on x. Formally, if $f\colon \mathbb{R}^m \mapsto \mathbb{R}$ is the linear objective function and $\{g_i\}_{i=1}^n$, where

*Under the supervision of Dr. Sutanu Roy, School of Mathematical Sciences, NISER

$g_i : \mathbb{R}^m \mapsto \mathbb{R}$ for all $1 \leq i \leq n$, are the linear constraints which are constrained by $\{b_i\}_{i=1}^n \subset \mathbb{R}$, then we are required to find a $x \in \mathbb{R}^m$ (or prove that no such vector exists) which

$$\begin{aligned} &\text{Maximizes (or minimizes) } f(x) \text{ subject to} \\ &g_1(x) \leq \ (\text{or} \geq \text{or} =)\ b_1 \\ &g_2(x) \leq \ (\text{or} \geq \text{or} =)\ b_2 \\ &\qquad\qquad \vdots \\ &g_n(x) \leq \ (\text{or} \geq \text{or} =)\ b_n \end{aligned} \tag{1.1}$$

Such a system is called a *linear program* or its abbreviation - *LP*. Given System 1.1, we can find a $b \in \mathbb{R}^n$, $c \in \mathbb{R}^m$, and $A \in \mathcal{M}(n, m)$ such that the following LP is equivalent to it.

$$\begin{aligned} &\text{Maximize (or minimize) } \langle c, x \rangle \text{ subject to} \\ &(Ax)_1 \leq \ (\text{or} \geq \text{or} =)\ b_1 \\ &(Ax)_2 \leq \ (\text{or} \geq \text{or} =)\ b_2 \\ &\qquad\qquad \vdots \\ &(Ax)_n \leq \ (\text{or} \geq \text{or} =)\ b_n \end{aligned} \tag{1.2}$$

In this report, we let X_j denote the set $\{x = (x_1, x_2 \dots x_j) \in \mathbb{R}^j \mid x_i \geq 0 \text{ for all } 1 \leq i \leq j\}$ for a $j \in \mathbb{N}$. For a vector $v \in \mathbb{R}^n$, we let v_i denote its i^{th} coordinate.

Definition (Canonical Linear Program). Given a $b \in \mathbb{R}^n$, $c \in \mathbb{R}^m$, and $A \in \mathcal{M}(n, m)$, find a $x \in X_m$ such that $(Ax)_i \leq b_i$ for all $1 \leq i \leq n$ which maximizes $\langle c, x \rangle$. That is, we are required to:

$$\begin{aligned} &\text{Maximize } \langle c, x \rangle \text{ subject to} \\ &(Ax)_i \leq b_i \text{ for all } 1 \leq i \leq n \\ &x \in X_m \end{aligned} \tag{1.3}$$

This system is called the *canonical* LP. In practice; b, c, and A are usually taken over $\mathbb{Q}$.

This LP is said to be in the *canonical form* since other linear programming

variations (a maximization problem, the variables taking values over all of $\mathbb{R}^m$, equalities in the constraints, and $\geq$ in the constraints) can be reduced to System 1.3 efficiently and quickly. A $x \in \mathbb{R}^m$ is called a *feasible solution* of System 1.3 if it satisfies all the constraints of that LP: that is, $x \in X_m$ and $(Ax)_i \leq b_i$ for all $1 \leq i \leq n$. If the solution space of a system is empty, then the system is called *infeasible*. We let OPT denote the optimal value of the objective function of a LP.

The first algorithm to solve the linear programming problem was given in 1947 by Dantzig and is called the *Simplex Algorithm*. Even though the algorithm solves most LPs efficiently, it has poor worst-case runtime - for a few cases, the algorithm runs in exponential time. It was only in 1979 that the linear programming problem was proven to be polynomial time solvable when Khachiyan introduced the *Ellipsoid Method*. There have been further improvements in running times of algorithms solving this problem. Currently, the best algorithms have running times which are slightly worse than quadratic times. The author refers the reader to [6] for detailed discussions of these algorithms.

We now shift our focus to the concept of *duality* of linear programs. Consider the following LP:

$$\begin{aligned}
&\text{Maximize } 5x_1 + 3x_2 + 4x_3 + x_4 \text{ subject to} \\
&4x_1 + x_2 + 0x_3 + x_4 \leq 6 \\
&2x_1 + x_2 + x_3 + 0x_4 \leq 4 \\
&x_1 + 0x_2 + x_3 + 0x_4 \leq 2 \\
&x_j \geq 0 \text{ for all } 1 \leq j \leq 4
\end{aligned}$$

Since we need to maximize the objective function, we find upper bounds for OPT and minimize these bounds. Formally, we try to minimize $y = (y_1, y_2, y_3) \in \mathbb{R}^3$ such that $y_1 \cdot (4x_1 + x_2 + 0x_3 + x_4) + y_2 \cdot (2x_1 + x_2 + x_3 + 0x_4) + y_3 \cdot (x_1 + 0x_2 + x_3 + 0x_4)$ is at least $5x_1 + 3x_2 + 4x_3 + x_4$. That is, we need to

$$\begin{aligned}
&\text{Minimize } 6y_1 + 4y_2 + 2y_3 \text{ subject to} \\
&4y_1 + 2y_2 + y_3 \geq 5 \\
&y_1 + y_2 + 0y_3 \geq 3 \\
&0y_1 + y_2 + y_3 \geq 4
\end{aligned}$$

$$y_1 + 0y_2 + 0y_3 \geq 1$$
$$y_i \geq 0 \text{ for all } 1 \leq i \leq 3$$

This LP is called the *dual* of the original LP (which is sometimes referred to as the *primal* LP). In general, given a canonical linear program as in System 1.3, its dual LP would be: find a $y \in X_n$ such that $(A^T y)_j \geq c_j$ for all $1 \leq j \leq m$ which minimizes $\langle b, y \rangle$. Equivalently, we are required to:

$$\begin{aligned} &\text{Minimize } \langle b, y \rangle \text{ subject to} \\ &(A^T y)_j \geq c_j \text{ for all } 1 \leq j \leq m \\ &y \in X_n \end{aligned} \tag{1.4}$$

System 1.4 can be rewritten as follows: given a $b \in \mathbb{R}^n$, $c \in \mathbb{R}^m$, and $A \in \mathcal{M}(n, m)$, find a $y \in X_n$ such that $(-A^T y)_j \geq -c_j$ for all $1 \leq j \leq m$ which maximizes $\langle -b, y \rangle$. Taking a dual of this system gives us the primal LP. It is natural to ask if the optimal values of the objective functions of the primal and the dual LP are the same. This is, in fact, true and is proved in Section 2 as the *Strong Duality Theorem*. The *Weak Duality Theorem*, stated and proved below, asserts that the optimal value of the objective function of the dual is at least that of the primal.

Theorem 1.1 (Weak Duality Theorem). Let x be a feasible solution of System 1.3 and y be a feasible solution of its dual, System 1.4. Then, $\langle b, y \rangle \geq \langle c, x \rangle$.

Proof. For any feasible solutions x and y of the primal and the dual respectively, we have:

$$\begin{aligned} \langle b, y \rangle &\geq \langle Ax, y \rangle && ((Ax)_i \leq b_i \text{ for all } 1 \leq i \leq n) \\ &= \langle x, A^T y \rangle && (A \text{ is a real matrix}) \\ &\geq \langle x, c \rangle && ((A^T y)_j \geq c_j \text{ for all } 1 \leq j \leq m) \\ &= \langle c, x \rangle && (c, x \text{ are vectors in } \mathbb{R}^n) \end{aligned}$$

This completes the proof of this theorem. □

Note that, in particular, if x^* is an optimal solution of System 1.3 and y^* is an optimal solution of its dual, System 1.4, then the *Weak Duality Theorem*

implies that $\langle b, y^* \rangle \geq \langle c, x^* \rangle$.

We will use the *Strong Duality Theorem* in the next subsection of the report. The theorem is stated formally below and is proved as Theorem 2.7 in Section 2.

Theorem 1.2 (Strong Duality Theorem). Let x^* be an optimal solution of System 1.3 and y^* be an optimal solution of its dual, System 1.4. Then, $\langle c, x^* \rangle = \langle b, y^* \rangle$.

We now state a useful corollary of the *Strong Duality Theorem*.

Definition. Let x be a feasible solution of System 1.3 and y be a feasible solution of its dual, System 1.4. Then, x and y are said to obey *complementary slackness conditions* if for all j, where $1 \leq j \leq m$, such that $x_j > 0$, $(A^T y)_j = c_j$ and for all i, where $1 \leq i \leq n$, such that $y_i > 0$, $(Ax)_i = b_i$.

Lemma 1.3 (Complementary Slackness). Let x be a feasible solution of System 1.3 and y be a feasible solution of its dual, System 1.4. Then, x and y obey complementary slackness conditions if, and only if, x and y are optimal solutions of their respective LPs.

Proof. Let $I = \{i \mid y_i > 0, 1 \leq i \leq n\}$ and $J = \{j \mid x_j > 0, , 1 \leq j \leq m\}$.

Assume that x and y obey complementary slackness conditions. Then,

$$\langle b, y \rangle = \sum_{i=1}^{n} b_i y_i = \sum_{i \in I} b_i y_i = \sum_{i \in I} (Ax)_i y_i = \sum_{i=1}^{n} (Ax)_i y_i = \langle Ax, y \rangle$$

Similarly,

$$\langle c, x \rangle = \sum_{j=1}^{m} c_j x_j = \sum_{j \in J} c_j x_j = \sum_{j \in J} (A^T y)_j x_j = \sum_{j=1}^{m} (A^T y)_j x_j = \langle A^T y, x \rangle$$

Since $\langle Ax, y \rangle = \langle A^T y, x \rangle$, $\langle b, y \rangle = \langle c, x \rangle$. Let x' and y' be feasible solutions of the primal and dual respectively. Then, by the *Weak Duality Theorem*,

$$\langle c, x \rangle = \langle b, y \rangle \geq \langle c, x' \rangle$$

Thus, x is an optimal solution of System 1.3. Similarly, we infer that y is an optimal solution of System 1.4.

Now, assume that x and y are the optimal solutions of System 1.3 and System 1.4 respectively. Then, by the *Strong Duality Theorem*, $\langle b, y\rangle = \langle c, x\rangle$. By the proof of the *Weak Duality Theorem*, this implies that $\langle b, y\rangle = \langle Ax, y\rangle$ and $\langle x, A^T y\rangle = \langle x, c\rangle$.

$$\langle b, y\rangle = \langle Ax, y\rangle \implies \sum_{i=1}^{n} b_i y_i = \sum_{i=1}^{n} (Ax)_i y_i \implies \sum_{i \in I} (b_i - (Ax)_i) y_i = 0$$

Since $(Ax)_i \leq b_i$ for all $1 \leq i \leq n$, this implies that for all $i \in I$, $b_i = (Ax)_i$. Similarly, we can prove that for all $j \in J$, $c_j = (A^T y)_j$. □

Definition. Let P_1 be a linear program. A LP P_2 is said to be a *relaxation* of P_1 if both P_1 and P_2 have the same objective function and every feasible solution of P_1 is a feasible solution of P_2. If P_1 and P_2 are maximization problems, then, clearly, $\text{OPT}_{P_1} \leq \text{OPT}_{P_2}$. If they are minimization problems, $\text{OPT}_{P_1} \geq \text{OPT}_{P_2}$.

Definition. Let α be a real number. An *α-approximation algorithm* for an optimization problem $\mathcal{P}$ is an algorithm which reports, for any instance I of $\mathcal{P}$, a value of at most $\alpha \cdot \text{OPT}$, where OPT is the value of the optimal solution of that instance. If $\mathcal{P}$ is a maximization problem, then $\alpha \leq 1$. Otherwise, $\alpha \geq 1$.

In the rest of this section we will look at two problems which have LP formulations - the Set Cover problem and the Max-Flow problem.

1.1 The Set Cover Problem

The goal of this subsection is to understand the Set Cover problem and how linear programming can be used to design fast approximation algorithms for computationally hard problems. We first define the Set Cover problem and then present a LP relaxation of the problem. We will then use this LP to design three rounding algorithms for Set Cover.

Problem (Set Cover). Let $E = \{e_1, e_2 \dots e_n\}$ be a set of elements and $\mathcal{S} = \{S_1, S_2, \dots S_m\} \subseteq \mathcal{P}(E)$. Let w, called the *weight function*, be a map from $\mathcal{S}$ to the set of positive reals. Define w_j, for all $1 \leq j \leq m$, as $w(S_j)$. A set $I \subseteq \{1, 2 \dots m\}$ is called a *set cover* of E if $\cup_{j \in I} S_j = E$. Find a set cover I' of E such that $\sum_{j \in I'} w_j \leq \sum_{j \in I} w_j$ for all set covers I of E.

If $w_j = 1$ for all $1 \leq j \leq m$, then this restricted version of the problem is called Unweighted Set Cover. The Unweighted Set Cover problem

can be reduced from the following important graph problems whose given graph instance is $G(V', E')$ where $V' = \{v_1, v_2 \dots v_n\}$:

- Vertex Cover: Let $S_j = \{e \in E' \mid e \text{ is incident on } v_j\}$ and E be the set of edges of G.
- Dominating Set: Let $S_j = N[v_j]$ (the closed neighbourhood of v_j) and E be the set of vertices.
- Clique Cover: Let $\mathcal{S}$ be the set of all cliques and E be the set of vertices.

The Set Cover problem can be modelled as follows: for each $S_j \in \mathcal{S}$, we have a corresponding "flag" variable which takes 1 if S_j is in the set cover and takes zero otherwise. To ensure that an element e_i of E is covered, we impose that at least one of the variables corresponding to a set containing e_i is 1. The objective is to minimize the weights of the sets in the set cover. Formally,

$$\begin{aligned} &\text{Minimize} \sum_{j=1}^{m} x_j w_j \text{ subject to} \\ &x_j \in \{0,1\} \text{ for all } 1 \leq j \leq m \\ &\sum_{j:e_i \in S_j} x_j \geq 1 \text{ for all } 1 \leq i \leq n \end{aligned} \tag{1.5}$$

Note that the above system is not a linear program. The following is a relaxation of the system defined previously:

$$\begin{aligned} &\text{Minimize} \sum_{j=1}^{m} x_j w_j \text{ subject to} \\ &x \in X_m \\ &\sum_{j:e_i \in S_j} x_j \geq 1 \text{ for all } 1 \leq i \leq n \end{aligned} \tag{1.6}$$

For an optimal solution x^* of System 1.6, $x_j^* \leq 1$ for all $1 \leq j \leq m$. Otherwise, if $x_j^* > 1$ for some such j, then replacing x_j^* by 1 in this solution is a feasible solution of this system whose objective functional value is strictly less than that of the original solution, a contradiction to our assumption of optimality. Thus, an vector corresponding to an optimal solution lies in $[0,1]^m$.

Definition. The *frequency* of an instance of the Set Cover problem, denoted by f, is the maximum number of times an element occurs in a set in $\mathcal{S}$ (for

example, in the VERTEX COVER problem, $f = 2$ since each edge is incident on exactly two vertices).

We now present three approximation algorithms to solve the SET COVER problem. The sketch of the algorithms are as follows: find an optimal solution of System 1.6 in polynomial time and construct a solution for SET COVER by carefully choosing a factor $\alpha \in [0, 1]$ and choosing only the sets corresponding to variables whose values are at least α in the set cover. These type of algorithms are called *rounding* algorithms. All the three algorithms presented have an approximation factor of f.

Algorithm 1.4 (Primal f-approximation Algorithm). Compute an optimal vector of System 1.6, say x^*. We construct a feasible solution of System 1.5 using the following procedure. Construct a vector $x' \in \mathbb{R}^m$ as follows:

$$x'_j = \begin{cases} 1 & \text{if } x^*_j \geq \frac{1}{f} \\ 0 & \text{if } x^*_j < \frac{1}{f} \end{cases}$$

Report the collection of sets corresponding to the vector x'. △

Theorem 1.5. Algorithm 1.4 is a f-approximation algorithm for SET COVER.

Proof. First, we prove that the vector reported by the algorithm is indeed a feasible solution of System 1.5. For each $e_i \in E$, there exists a $S_j \in \mathcal{S}$ whose corresponding variable takes a value greater than or equal to $\frac{1}{f}$. Otherwise, x^* will fail to satisfy the constraint corresponding to e_i in System 1.6, a contradiction. This implies that x' is a feasible solution of System 1.5.

Now, we prove that the procedure described above is a f-approximation algorithm. Let $j \in \mathbb{N}$ such that $1 \leq j \leq m$.

If $x^*_j < \frac{1}{f}$, $x'_j = 0$. Thus, $x'_j \leq f \cdot x^*_j$. If $\frac{1}{f} \leq x^*_j$, $x'_j = 1 \leq f \cdot x^*_j$. This implies that $x'_j \leq f \cdot x^*_j$ for all $1 \leq j \leq m$.

Let OPT be the optimal solution of System 1.5 and OPT$'$ be the optimal solution of System 1.5. Clearly, OPT$' \leq$ OPT.

$$\sum_{j=1}^{m} x'_j w_j \leq \sum_{j=1}^{m} f \cdot x^*_j w_j \qquad (x'_j \leq f \cdot x^*_j \text{ for all } 1 \leq j \leq m)$$

$$
\begin{aligned}
&\leq f \cdot \sum_{j=1}^{m} x_j^* w_j \\
&= f \cdot \text{OPT}' && (x^* \text{ is an optimal solution}) \\
&\leq f \cdot \text{OPT} && (\text{OPT}' \leq \text{OPT})
\end{aligned}
$$

Thus, Algorithm 1.4 is a f-approximation algorithm for the Set Cover problem. □

We will now present two approximation algorithms that involve the dual of the Set Cover problem. Each constraint of System 1.6 forces the sum of the variables corresponding to the sets containing an element to be at least 1. Thus, each constraint in the dual will ensure that the sum of the variables corresponding to the elements of a set does not exceed the weight of the set itself. Formally, the dual of the Set Cover problem will be:

$$
\begin{aligned}
&\text{Maximize} \sum_{i=1}^{n} y_i \text{ subject to} \\
&y \in X_n \\
&\sum_{i:e_i \in S_j} y_i \leq w_j \text{ for all } 1 \leq j \leq m
\end{aligned}
\tag{1.7}
$$

By the *Weak Duality Theorem*, $\sum_{j=1}^{m} x_j w_j \geq \sum_{i=1}^{n} y_i$ for any feasible solutions x and y of Systems 1.6 and 1.7 respectively. Thus, $\text{OPT}' \geq \sum_{i=1}^{n} y_i$.

Algorithm 1.6 (Dual f-approximation Algorithm I). Compute an optimal solution of System 1.7, say y^*. Let I' be the collection of sets whose weights are equal to the sum of the values of the variables corresponding to the elements in that set. That is, $I' = \{S_j \mid \sum_{i:e_i \in S_j} y_i^* = w_j\}$. Report I'. △

Theorem 1.7. Algorithm 1.6 is a f-approximation algorithm for Set Cover.

Proof. We prove that I' is a set cover of E. Consider an element $e_k \in E$. Let $J = \{j \mid e_k \in S_j\}$. Define $\epsilon = \min_{j \in J}\{w_j - \sum_{i:e_i \in S_j} y_i^*\}$ where y^* is the optimal solution used in Algorithm 1.6 to produce I'. As imposed by the constraints of System 1.7, $\epsilon \geq 0$. Define a vector $y' \in \mathbb{R}^n$ as follows: $y'_k = y_k^* + \epsilon$ and $y'_i = y_i^*$ for all $i \neq k$, $1 \leq i \leq n$.

For any $j \notin J$,

$$\sum_{i:e_i \in S_j} y'_i = \sum_{i:e_i \in S_j} y^*_i \leq w_j$$

For any $j \in J$,

$$\sum_{i:e_i \in S_j} y'_i = \epsilon + \sum_{i:e_i \in S_j} y^*_i \leq (w_j - \sum_{i:e_i \in S_j} y^*_i) + \sum_{i:e_i \in S_j} y^*_i \leq w_j$$

Thus, for all $1 \leq j \leq m$, $\sum_{i:e_i \in S_j} y_i \leq w_j$. Clearly, $y' \in X_n$. This implies that y' is a feasible solution of System 1.7 and $\sum_j y'_j = \sum_j y^*_j + \epsilon$. Since y^* was an optimal solution of this system, $\epsilon = 0$. This implies that there exists a set S_j containing e_k such that $\sum_{i:e_i \in S_j} y^*_i = w_j$. Thus, I' is a set cover of E.

Let OPT be the optimal value of the SET COVER problem (System 1.5) and OPT$'$ be the optimal value of its linear programming relaxation (System 1.6). If x^* is an optimal solution of System 1.6, then, we have

$$\begin{aligned}
f \cdot \text{OPT} &\geq f \cdot \text{OPT}' && (\text{OPT} \geq \text{OPT}') \\
&= f \cdot \sum_{j=1}^{m} (x^*_j w_j) \\
&\geq f \cdot \sum_{i=1}^{n} y^*_i && (\textit{Weak Duality Theorem}) \\
&\geq \sum_{S_j \in \mathcal{S}} \sum_{i:e_i \in S_j} y^*_i && (\text{Definition of } f) \\
&\geq \sum_{S_j \in I'} \sum_{i:e_i \in S_j} y^*_i && (I' \subseteq \mathcal{S}) \\
&= \sum_{j:S_j \in I'} w_j && (\text{Construction of } I')
\end{aligned}$$

Thus, this algorithm returns a f-approximate solution as well. □

Let I be the set cover returned by Algorithm 1.4. Then, by construction, for all $S_j \in I$, $x^*_j \geq \frac{1}{f} > 0$ (where x^* is the optimal solution of System 1.6 considered by the algorithm). By Lemma 1.3, for all such S_j, $\sum_{i:e_i \in S_j} y^*_i = w_j$. Thus, $S_j \in I'$. This implies that $I \subseteq I'$. That is, Algorithm 1.4 does at least as well Algorithm 1.6 for any instance of the SET COVER problem.

Using the proof of Theorem 1.7, we can produce an algorithm for SET COVER

which does not include solving a LP.

Algorithm 1.8 (Dual f-approximation algorithm II). We report the set cover as given by the following procedure.

- $y \in \mathbb{R}^n$, $y \leftarrow 0$
- $I' \leftarrow \emptyset$
- **while** I' is not a set cover **do**
  - Find $e_l \in E$ which is not covered by I'
  - $J \leftarrow \{j \mid e_k \in S_j\}$
  - $\epsilon \leftarrow \min_{j \in J}\{w_j - \sum_{i:e_i \in S_j} y_i\}$
  - $y_l \leftarrow y_l + \epsilon$
  - $I' \leftarrow \{S_j \mid \sum_{i:e_i \in S_j} y_i = w_j\}$
- Report I' △

Lemma 1.9. Algorithm 1.8 is a f-approximation algorithm for Set Cover.

Proof. Let $k \in \mathbb{N}$. Denote the set that is produced in the k^{th} step of this algorithm be I'_k and the dual vector be $y^{(k)}$. Then, by definition, $I'_k = \{S_j \mid \sum_{i:e_i \in S_j} y_i^{(k)} = w_j\}$. If I'_k is a set cover of E and report it if it is. If I'_k is not a set cover, then, there exists an $e_l \in E$ which is not covered by I'_k. Thus, over all the S_j which contains e_l, $\epsilon = \min_j\{w_j - \sum_{i:e_i \in S_j} y_i^{(k)}\} > 0$ (by the proof of correctness of Algorithm 1.6). $y^{(k+1)}$ is defined as follows: $y_l^{(k+1)} = y_l^{(k)} + \epsilon$ and $y_i^{(k+1)} = y_i^{(k)}$ for all $i \neq l$. Clearly, $y^{(k+1)}$ is still a feasible solution of System 1.7 and I_{k+1} contains e_l.

This implies that we will need to run this algorithm for at most n many steps since at each step we cover at least one new element. Assume that the algorithm runs p times. Then, I'_p will be a f-approximate solution as we can apply the same proof as used in the correctness of Algorithm 1.6 (replace y^* by $y^{(p)}$ and I' by I'_p in that proof). □

1.2 The Maximum Flow Problem

In this subsection, we will study the Max-Flow and Min-Cut problems to better understand the concept of duality. We will prove that the both of these

problems admit linear programming formulations and that they are the duals of each other.

Let $(G(V, E))$ be a directed graph with two special vertices s (called the *source*) and t (called the *sink*) whose indegree and outdegree respectively is 0. The *capacity*, denoted by c, is a function from the set of arcs, E, of this graph to the set of non-negative reals. The *capacity of an edge* $e \in E$ is $c(e)$. A *network* is a tuple $(G(V, E), c)$ where c is a capacity function of the directed graph $G(V, E)$ which has a source and a sink. A *flow* is a function $f : E \mapsto \mathbb{R}_+$ which satisfies the following constraints:

(i) $f((u, v)) \leq c((u, v))$ for all $(u, v) \in E$.

(ii) $\sum_{v \in N(u)} f((u, v)) = \sum_{w:u \in N(w)} f((w, u))$ for all $u \in V \setminus \{s, t\}$.

For the sake of simplicity, we abuse notation to rewrite the second constraint by summing over all $v \in V$ instead of the neighbours of u. It is intrinsically understood that the flow values of the other tuples are 0. Thus, we can rewrite the constraints as follows:

(i) $f((u, v)) \leq c((u, v))$ for all $(u, v) \in E$.

(ii) $\sum_{v \in V} f((u, v)) = \sum_{w \in V} f((w, u))$ for all $u \in V \setminus \{s, t\}$.

The *cost of the flow* f is defined to be the $\sum_{v \in V} f((s, v))$.

Problem (Maximum Flow). Given a network $(G(V, E), c)$, find a flow f' of this network which has the maximum cost.

We prove that the optimal flow always exists.

Lemma 1.10. Given a network $(G(V, E), c)$, there exists a flow f' such that for all flows f of this network, $\sum_{v \in V} f'(s, v) \geq \sum_{v \in V} f(s, v)$.

Proof. Let $E = \{e_1, e_2 \dots e_m\}$. We can think of any flow f as a m-tuple: $(f(e_1), f(e_2) \dots f(e_m))$. Let $S \subseteq \mathbb{R}^m$ be the set of all flows defined on the network. Since $0 \in S$, $S \neq \emptyset$. For any $f \in S$, $0 \leq f(e_j) \leq c(e_j)$ for each j such that $1 \leq j \leq m$. Furthermore, the flow conservation constraints are of the form:

$$\sum_{j=1}^{m} a_i f(e_i) = 0$$

where $a_i \in \{-1, 0, 1\}$. Thus, S is the intersection of finitely many hyperplanes

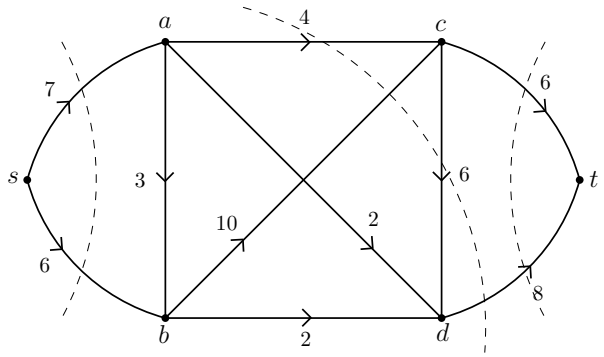


Figure 1: A network with cuts.

and finitely many halfspaces. This implies that it is closed. Since S is clearly bounded, it is compact. Define a function $g : S \mapsto \mathbb{R}$ as follows

$$g(f(e_1), f(e_2) \dots f(e_m)) : = \sum_{v \in V} f(s, v)$$

Clearly, g is continuous. Since S is compact, g attains it maximum at some point which corresponds to a flow, say f'. That is, for all the flows f in the solution space,

$$g(f'(e_1), f'(e_2) \dots f'(e_m)) \geq g(f(e_1), f(e_2) \dots f(e_m))$$

This implies that $\sum_{v \in V} f'(s, v) \geq \sum_{v \in V} f(s, v)$ for all flows f, completing the proof of the lemma. □

Consider the network illustrated in Figure 1. Since we are trying to maximise the cost of the flow through this network, we try to find an upper bound for the flow, and then look to make this upper bound as tight as possible. Since the flow has to "reach" t, it has to be constrained by the sum of the capacities of the arcs incident on t. Thus, we get our first upper bound: $6 + 8 = 14$. Similarly, it is bounded by the sum of the capacities of the arcs emerging from s. This gives us a better bound: $7 + 6 = 13$. It is natural to expect that it is bounded by the sum of the capacities of edges going from a set containing s to a set containing t. In our example, it will be bounded by $4+8 = 12$. We now prove this formally. First, we define a few terms and prove a useful theorem.

Definition. A *cut* of the graph is any subset A of V containing s and not

containing t. The *cost of the cut*, $c(A)$, to be defined as follows:

$$c(A) = \sum_{u \in A} \sum_{v \notin A} c((u, v))$$

while the *net flow out of the cut*, $f(A)$, is as follows:

$$f(A) = \sum_{u \in A} \sum_{v \notin A} f((u, v)) - \sum_{v \notin A} \sum_{u \in A} f((v, u))$$

Problem (Minimum Cut). Given a network $(G(V, E), c)$, find a cut A of this network which has the minimum cost.

Since there are finitely many cuts ($\mathcal{O}(2^{|V|})$ many), there exists a cut with the minimum cost.

Theorem 1.11. Let f be a flow of a network $(G(V, E), c)$ and A be a cut of G. Then, the net flow out of the cut is equal to the cost of the flow.

Proof. Let $A' = A \setminus \{s\}$. For any $u \in A'$, we have

$$\begin{aligned}
&\sum_{v \in V} f((u, v)) = \sum_{w \in V} f((w, u)) \\
\implies &\sum_{v_1 \in A} f((u, v_1)) + \sum_{v_2 \notin A} f((u, v_2)) = \sum_{w_1 \in A} f((w_1, u)) + \sum_{w_2 \notin A} f((w_2, u)) \\
\implies &\sum_{v_2 \notin A} f((u, v_2)) - \sum_{w_2 \notin A} f((w_2, u)) = \sum_{w_1 \in A} f((w_1, u)) - \sum_{v_1 \in A} f((u, v_1))
\end{aligned}$$

Summing over all such u, the left-hand side of last equality becomes:

$$\sum_{u \in A'} \sum_{v_2 \notin A} f((u, v_2)) - \sum_{w_2 \notin A} \sum_{u \in A'} f((w_2, u)) = f(A) - \sum_{v_2 \notin A} f((s, v_2))$$

The right-hand side becomes:

$$\begin{aligned}
&\sum_{u \in A'} \Big(\sum_{w_1 \in A} f((w_1, u)) - \sum_{v_1 \in A} f((u, v_1)) \Big) \\
&= \sum_{u \in A'} \Big(\sum_{w_1 \in A'} f((w_1, u)) - \sum_{v_1 \in A} f((u, v_1)) \Big) + \sum_{v_1 \in A} f((s, v_1)) \\
&= \sum_{u \in A'} \Big(\sum_{w_1 \in A'} f((w_1, u)) - \sum_{v_1 \in A'} f((u, v_1)) \Big) + \sum_{v_1 \in A} f((s, v_1))
\end{aligned}$$

$$= \sum_{u \in A'} \sum_{w_1 \in A'} f((w_1, u)) - \sum_{u \in A'} \sum_{v_1 \in A'} f((u, v_1)) + \sum_{v_1 \in A} f((s, v_1))$$
$$= \sum_{v_1 \in A} f((s, v_1))$$

Equating the two sides, we get:

$$f(A) - \sum_{v_2 \notin A} f((s, v_2)) = \sum_{v_1 \in A} f((s, v_1)) \implies f(A) = \sum_{v \in V} f((s, v))$$

This completes the proof of this theorem. □

Lemma 1.12. Let f be a flow of a network $(G(V, E), c)$ and A be a cut of G. Then, the cost of the flow is at most the cost of the cut.

Proof. By Theorem 1.11, we have

$$\begin{aligned}
\sum_{v \in V} f((s, v)) &= f(A) \\
&= \sum_{u \in A} \sum_{v \notin A} f((u, v)) - \sum_{v \notin A} \sum_{u \in A} f((v, u)) \\
&\leq \sum_{u \in A} \sum_{v \notin A} f((u, v)) \\
&\leq \sum_{u \in A} \sum_{v \notin A} c((u, v)) \\
&\leq c(A)
\end{aligned}$$

This completes the proof of the lemma. □

If there exists a flow f' whose cost is equal to the cost of a cut, then this cut must have the minimum cost and the flow must be optimal. We now prove there indeed exists a cut whose cost is the same as the cost of an optimal flow.

Theorem 1.13. Let f' be an optimal flow of a network $(G(V, E), c)$. There exists a cut A of this network such that $c(A) = \sum_{v \in V} f'((s, v))$.

Proof. We construct a network $(G'(V, V^2), c')$ where the capacity function c' is defined as follows:

$$c'((u, v)) = c((u, v)) - f'((u, v)) + f'((v, u)) \tag{1.8}$$

Note that c' is indeed a capacity function of G'. Let A be the set of vertices in V such that there exists a path p from s to v in G' such that $c'(e) > 0$ for all $e \in p$. Clearly, $s \in A$. $t \notin A$ as otherwise f' would not be the maximum flow. Thus, A is a cut of G. By Theorem 1.11, we have

$$\sum_{v \in V} f'((s,v)) = f'(A) = \sum_{u \in A, v \notin A} f'((u,v)) - \sum_{v \notin A, u \in A} f'((v,u)) \tag{1.9}$$

Consider $f'((u,v))$ where $u \in A$ and $v \notin A$. Since $u \in A$, there exists a path p from s to u such that $c'(e) > 0$ for all $e \in p$. If $c'((u,v)) > 0$, then $v \in A$ since $p \cup \{(u,v)\}$ would be a path from s to v containing edges with non-zero capacity values. This is a contradiction to our initial assumption that $v \notin A$. Thus, $c'((u,v)) = 0$. Substituting for this term in Equation (1.8), we get $f'((u,v)) = c((u,v)) + f'((v,u))$. Thus, Equation (1.9) becomes

$$\begin{aligned}
\sum_{v \in V} f'((s,v)) &= \sum_{u \in A} \sum_{v \notin A} f'((u,v)) - \sum_{v \notin A} \sum_{u \in A} f'((v,u)) \\
&= \sum_{u \in A} \sum_{v \notin A} c((u,v)) + \sum_{u \in A} \sum_{v \notin A} f'((v,u)) - \sum_{v \notin A} \sum_{u \in A} f'((v,u)) \\
&= c(A)
\end{aligned}$$

As observed after the proof of Lemma 1.12, this implies that A is an optimal cut and that the minimum cost of a cut is exactly the same as the value of the maximum flow in a network. □

In the rest of the section, we will prove that the MAX-FLOW problem can be modeled as a linear program and its dual is indeed the MIN-CUT problem.

The LP of the MAX-FLOW problem is straightforward to derive from its definition:

$$\begin{aligned}
&\text{Maximize} \sum_{v \in V} f((s,v)) \text{ subject to} \\
&f(e) \geq 0 \text{ for all } e \in E \\
&f(e) \leq c(e) \text{ for all } e \in E \\
&\sum_{u \in V} f((u,v)) = \sum_{w \in V} f((v,w)) \text{ for all } v \in V \setminus \{s,t\}
\end{aligned} \tag{1.10}$$

We will construct the dual of this LP by constructing an equivalent LP (System

1.11) and taking its dual (System 1.12). We will finally prove that System 1.12 is indeed equivalent to the MIN-CUT problem in Theorem 1.17. We first prove a few lemmas to help with these constructions.

Lemma 1.14. Let f be a flow of a network $(G(V, E), c)$ such that all the paths from s to t have an edge e in them such that $f(e) = 0$. Then, the cost of the flow is 0.

Proof. Let A be the set of vertices that can be reached using a path from s such that the flow through each edge of the path is greater than 0. Then, $s \in A$ and $t \notin A$ implying that A is a cut. $f((u, v)) = 0$ for all $u \in A$ and $v \notin A$ as otherwise v would be in A. We have,

$$f(A) = \sum_{u \in A} \sum_{v \notin A} f((u, v)) - \sum_{v \notin A} \sum_{u \in A} f((v, u)) \leq \sum_{u \in A} \sum_{v \notin A} f((u, v)) \leq 0$$

By Theorem 1.11,

$$\sum_{v \in V} f((s, v)) = f(A) \leq 0 \implies \sum_{v \in V} f((s, v)) = 0$$

This completes the proof of the lemma. □

Theorem 1.15. Given any flow f of a network $(G(V, E), c)$, there exists k-many flows $f_1, f_2 \dots f_k$ on this network, corresponding to k-many paths $p_1, p_2 \dots p_k$ from s to t, such that:

(i) $k \leq |E|$.

(ii) $f_i((u, v)) > 0$ if, and only if $(u, v) \in p_i$ for all $1 \leq i \leq k$.

(iii) The sum of costs of the f_i's is the cost of f.

(iv) $\sum_{i=1}^{k} f_i((u, v)) \leq f((u, v))$ for all $(u, v) \in E$.

Proof. We construct the f_i's by using Lemma 1.14 by the following procedure:

- $r \colon E \mapsto \mathbb{R}$
- $r \leftarrow f$
- $i \leftarrow 1$
- **while** the cost of $r > 0$ **do**

- Find a path p_i from s to t such that $r(e) > 0$ for all $e \in p_i$
- Define $f_i \colon E \mapsto \mathbb{R}$ as follows: $f_i(e') = \min_{e \in p_i} r(e)$ for all the $e' \in p_i$ and $f_i(e') = 0$ otherwise
- $r(e) \leftarrow r(e) - f_i(e)$ for each $e \in E$
- $i \leftarrow i + 1$

Claim 1.15.1. Let $r^{(l)}$ be the function r in the l^{th} iteration of this procedure. Then, for any $l \in \mathbb{N}$, $r^{(l)}$ is a flow.

Proof. We prove this by induction on l. Clearly, $r^{(1)} = f$ is a flow. Now, assume that $r^{(l-1)}$ is a flow. By construction, $f_{l-1}(e) \leq r^{(l-1)}(e)$ for all $e \in p_{l-1}$ and $f_{l-1}(e) = 0 \leq r^{(l-1)}(e)$ for all $e \notin p_{l-1}$. Thus, $r^{(l)}(e) \geq 0$ for all $e \in E$. Also, clearly, $r^{(l)}(e) \leq r^{(l)}(e) \leq c(e)$ for all $e \in E$. Let $u \in V \setminus \{s, t\}$. If u is not incident on any $e \in p_l$, then

$$\sum_{v \in V} r^{(l)}((u,v)) = \sum_{v \in V} r^{(l-1)}((u,v)) = \sum_{v \in V} r^{(l-1)}((v,u)) = \sum_{v \in V} r^{(l)}((v,u))$$

Assume that u is incident on an edge $e_1 = (u, v_1) \in p_l$. Then, the only other edge u is incident on is the previous edge describing p_l, say $e_2 = (v_2, u)$. Then, since $r^{(l)}((u,v)) = r^{(l-1)}((u,v_1)) - f_l((u,v_1))$, we have,

$$\begin{aligned}
\sum_{v \in V} r^{(l)}((u,v)) &= \sum_{v \in V} r^{(l-1)}((u,v)) - f_l((u,v_1)) \\
&= \sum_{v \in V} r^{(l-1)}((v,u)) - f_l((u,v_1)) \qquad \text{(Induction hypothesis)} \\
&= \sum_{v \in V} r^{(l-1)}((v,u)) - f_l((v_2,u)) \quad (f_l((u,v_1)) = f_l((v_2,u))) \\
&= \sum_{v \in V} r^{(l)}((v,u))
\end{aligned}$$

Thus, $r^{(l)}$ remains a flow throughout the course of this procedure. ◇

Note that this procedure is indeed an algorithm since in an iteration l, $r^{(l)}(e)$ is 0 for at least one edge $e \in E$ with $r^{(l-1)}(e) > 0$. Thus, the algorithm runs k-many times where $k \leq |E|$. Also, note that at each iteration of the algorithm, we can find a path from s to t called p_l such that $r^{(l)}(e) > 0$ for all $e \in p_l$ by a direct consequence of Lemma 1.14. It is also clear that the f_is that are constructed are indeed flows for all $1 \leq i \leq k$.

This proves that this collection of flows and paths satisfy the first two parts of the theorem. Since the algorithm runs until the cost of r drops to 0 and it reduces the cost of r by the cost of f_i at each step i, the third part of the theorem is also satisfied by these flows and paths. Part (iv) of the theorem is true since the algorithm removes the minimum flow through an edge of the chosen path in each iteration. □

We will use Theorem 1.15 to prove that the value of the optimal values of the objective functions of Systems 1.10 and 1.11, which follows, are the same. Even though these two LPs are not equivalent in the strictest sense of the definition, we prove that using the decomposition of a flow into paths described in Theorem 1.15, we can identify a flow with a set of paths and the vice versa. We abuse our definition of equivalence and call these LPs equivalent.

Let $\mathcal{P}$ denote the set of all paths from s to t in a network $(G(V, E), c)$.

$$\begin{aligned} &\text{Maximize} \sum_{p \in \mathcal{P}} x_p \text{ subject to} \\ &x_p \geq 0 \text{ for all } p \in \mathcal{P} \\ &\sum_{p : e \in p} x_p \leq c(e) \text{ for all } e \in E \end{aligned} \tag{1.11}$$

Lemma 1.16. System 1.10 and System 1.11 are equivalent linear programs.

Proof. Let f be a feasible solution of System 1.10. Then, by Theorem 1.15, there exists k-many flows, where $k \leq |E|$, $f_1, f_2 \dots f_k$ corresponding to paths $p_1, p_2 \dots p_k$ which are from s to t. For $p \in \mathcal{P}$, define x_p as follows:

$$x_p = \begin{cases} \text{cost of } f_p & \text{if } p \in \{p_1, p_2 \dots p_k\} \\ 0 & \text{if } p \notin \{p_1, p_2 \dots p_k\} \end{cases}$$

By Theorem 1.15, $\{x_p\}_{p \in \mathcal{P}}$ is a feasible solution of System 1.11 and $\sum_{p \in \mathcal{P}} x_p = \sum_{v \in V} f((s, v))$.

Let $\{x_p\}_{p \in \mathcal{P}}$ be a feasible of System 1.11. Define $f \colon E \mapsto \mathbb{R}$ as follows:

$$f(e) = \sum_{p : e \in p} x_p$$

Clearly, $0 \leq f(e) \leq c(e)$ for all $e \in E$. For some $v \in V \setminus \{s, t\}$, note that

$$\begin{aligned}\sum_{u \in V} f((u, v)) &= \sum_{u \in V} \sum_{p:(u,v) \in p} x_p \\ &= \sum_{p:v \in V(p)} x_p \\ &= \sum_{w \in V} \sum_{p:(v,w) \in p} x_p \\ &= \sum_{w \in V} f((v, w))\end{aligned}$$

Thus, the f we defined is indeed a feasible solution of System 1.10. This proves that Systems 1.10 and 1.11 are equivalent. □

Constructing the dual of System 1.11 is straightforward:

$$\begin{aligned}&\text{Minimize} \sum_{e \in E} y_e c(e) \text{ subject to} \\ &y_e \geq 0 \text{ for all } e \in E \\ &\sum_{e \in p} y_e \geq 1 \text{ for all } p \in \mathcal{P}\end{aligned} \tag{1.12}$$

We prove that the optimal solution of System 1.12 is equal to the capacity of the minimum cut in the theorem that follows.

Theorem 1.17. The optimal solution of System 1.12 is equal to that of the Min-Cut problem.

Proof. First, we prove that an arbitrary cut A of a network $(G(V, E), c)$ corresponds to a feasible solution of System 1.12. Let $y_{(u,v)} = 1$ for all $u \in A$, $v \notin A$ and 0 for all other $y_{(u,v)}$. It is easy to see, by induction on the length of a path $p \in \mathcal{P}$, that there exists a edge $(u, v) \in p$ such that $u \in A$ and $v \notin A$. Thus, $\sum_{e \in p} y_e \geq 1$ for any path $p \in \mathcal{P}$. This proves that $\{y_e\}_{e \in E}$ is a feasible solution of System 1.12. Thus, the optimal solution of System 1.12 is bounded above by the capacity of the minimal cut.

To complete this proof, given any feasible solution $\{y_e\}_{e \in E}$ of System 1.12, we produce a cut whose capacity is at most $\sum_{e \in E} y_e c(e)$. For a $v \in V$, let $\mathcal{P}_v$ be

the set of paths from s to v. Note that $\mathcal{P}_t = \mathcal{P}$. Define $d\colon V \mapsto \mathbb{R}$ as follows:

$$d(v) = \min_{p \in \mathcal{P}_v} \sum_{e \in p} y_e$$

If $(u, v) \in E$, then,

$$d(v) \leq d(u) + y_{(u,v)} \implies d(v) - d(u) \leq y_{(u,v)} \tag{1.13}$$

Pick a $t \in [0, 1)$ universally at random. Let $A = \{v \mid d(v) \leq t\}$. Clearly, $s \in A$ and $t \notin A$ implying that A is a cut of G. Thus,

$$c(A) = \sum_{u \in A, v \notin A} c((u, v)) = \sum_{(u,v) | d(u) \leq t < d(v)} c((u, v))$$

The expected value of the capacity can be calculated as follows:

$$\begin{aligned}
\mathbb{E}_{T \sim U[0,1)} c(A) &= \sum_{(u,v) \in E} c((u, v)) \cdot \mathbf{Pr}[d(u) \leq t < d(v)] \\
&= \sum_{(u,v) \in E} c((u, v)) \cdot (d(v) - d(u)) \\
&\leq \sum_{(u,v) \in E} y_{(u,v)} c((u, v)) \qquad \text{(Equation (1.13))}
\end{aligned}$$

Thus, there exists a cut, say A', such that $c(A') \leq \sum_{e \in E} y_e c(e)$. Which implies the optimal solution of System 1.12 is bounded below by the capacity of the minimal cut. This proves that the optimal solution of System 1.12 is exactly the capacity of a minimal cut. □

By the *Strong Duality Theorem*, this is exactly the optimal solution of the objective function of System 1.11 and thus, by Lemma 1.16, of System 1.10.

2 The Strong Duality Theorem

In this section, we prove the *Strong Duality Theorem* in Theorem 2.7. To prove this theorem, we first prove the following lemmas from analysis.

Lemma 2.1 (Weierstrass' Theorem). Let $X \subseteq \mathbb{R}^n$ be a compact set and $f : X \mapsto \mathbb{R}$ be a continuous function. Then, there exists a $x_0 \in X$ such that $f(x_0) \leq f(x)$ for all $x \in X$.

Proof. Since X is compact and f is continuous, $f(X)$ is compact. Thus, it is closed and bounded. This implies that there exists a $x_0 \in X$ where f attains its minimum value. □

Lemma 2.2 (Projection Lemma). Let $X \subseteq \mathbb{R}^n$ be a non-empty, closed, convex set and y be a point outside X. Then, there exists a $x_0 \in X$ such that $d(y, x_0) \leq d(y, x)$ for all $x \in X$. Here, $d(u, v) := ||u - v||_2$ for all $u, v \in \mathbb{R}^n$. Furthermore, for all $x \in X$, $\langle y - x_0, x - x_0 \rangle \leq 0$.

Proof. Since $X \neq \emptyset$, there is a $x' \in X$. Let $r = d(y, x')$. Let $\overline{B_r}(y)$ denote the closed ball of radius r around y and let $X' = X \cap \overline{B_r}(y)$. Clearly, X' is closed and bounded. Thus, it is compact. Furthermore, the minimum value of the distance function, if attained, must be attained inside X'. Since $f \colon X' \mapsto \mathbb{R}$ defined by $f(x) = d(y, x)$ is a continuous function, by *Weierstrass' Theorem*, there exists an $x_0 \in X'$ which minimizes f in X'. Thus, x_0 minimizes the distance between y and the set X.

Now, since X is convex, for all $x \in X$ and $\alpha \in (0, 1)$, $x_0 + \alpha(x - x_0) \in X$. Since $d(y, x_0) \leq d(y, x')$ for all $x' \in X$, we have,

$$\begin{aligned} ||y - x_0||^2 &\leq ||y - x_0 - \alpha(x - x_0)||^2 \\ &\leq ||y - x_0||^2 + \alpha^2 ||x - x_0||^2 - 2\alpha \langle y - x_0, x - x_0 \rangle \end{aligned}$$

Thus,

$$\langle y - x_0, x - x_0 \rangle \leq \frac{\alpha}{2} ||x - x_0||^2$$

As α can be as small as we require, $\langle y - x_0, x - x_0 \rangle \leq 0$. □

Lemma 2.3 (Hyperplane Separation Lemma). Let $X \subseteq \mathbb{R}^n$ be a non-empty, closed, convex set and $y \notin X$. Then, there exists a hyperplane $H = \{h \mid \langle a, h \rangle = \gamma\}$ with $a \in \mathbb{R}^n$ and $\gamma \in \mathbb{R}$ such that $\langle x, a \rangle \geq \gamma$ for all $x \in X$ and $\langle y, a \rangle < \gamma$.

Proof. By the *Projection Lemma*, there is a $x_0 \in X$ such that $\langle y - x_0, x - x_0 \rangle \leq 0$ for all $x \in X$. Let $a = x_0 - y$ and $\gamma = \langle a, x_0 \rangle$. Then, for a $x \in X$,

$$\begin{aligned} 0 &\geq \langle y - x_0, x - x_0 \rangle \\ &= \langle -a, x - x_0 \rangle \end{aligned}$$

$$= \langle -a, x \rangle + \langle a, x_0 \rangle$$
$$= \langle -a, x \rangle + \gamma$$

Thus, $\langle a, x \rangle \geq \gamma$. Also,

$$\begin{aligned}
\langle a, y \rangle &= \langle a, x_0 - a \rangle \\
&= \langle x_0 - y, x_0 \rangle - \langle a, a \rangle \\
&= \gamma - ||a||^2 \\
&= \gamma - ||x_0 - y||^2 \\
&< \gamma \qquad\qquad (y \neq x_0)
\end{aligned}$$

Thus, the hyperplane $H = \{h \mid \langle a, h \rangle = \gamma\}$ separates X and y. □

As defined previously, we let X_j denote the set $\{x = (x_1, x_2 \dots x_j) \in \mathbb{R}^j \mid x_i \geq 0 \text{ for all } 1 \leq i \leq j\}$ for a $j \in \mathbb{N}$.

Theorem 2.4. Let $A \colon \mathbb{R}^p \mapsto \mathbb{R}^q$ be a linear map. Then, $A(X_p)$ is closed.

Proof. Let $\{y_n\}$ be a sequence in $A(X_p)$ which converges to a point $y \in \mathbb{R}^q$. We prove that $y \in A(X_p)$. Let $I_n = \{x \in X_p \mid A(x) = y_n\}$. Since X_p is closed and $A^{-1}(\{y_n\})$ is closed, I_n is closed. Also, by definition, it is not empty. Since $|| \cdot ||_2$ is bounded below and is continuous, there exists a $x_n \in I_n$ such that

$$||x_n|| = \inf_{x \in I_n} ||x||$$

If $\{x_n\}$ is bounded, then it has a subsequence $\{x_{n_k}\}$ such that $\{x_{n_k}\}$ converges to some $x' \in \mathbb{R}^p$. Since each $x_{n_k} \in X$, $x' \in X_p$ as X_p is closed. Also, by definition $A(x_{n_k}) = y_{n_k}$. This implies that $A(x') = y$ since $\{y_{n_k}\}$ converges to y and A is continuous. Thus, $y \in A(X_p)$.

Now, assume that $\{x_n\}$ is unbounded. We prove that this contradicts our assumption that x_n has the least norm in I_n for all $n \in \mathbb{N}$.

Since $||x_n|| \to \infty$, there exists $M_0 \in \mathbb{N}$ such that $||x_n|| > 1$ for all $n \geq M_0$. For all such n, define unit vectors z_n by $z_n = \frac{x_n}{||x_n||}$. Since $\{z_n\}$ is bounded, there

exists $\{z_{n_k}\}$, a subsequence, which converges to some unit vector $z \in \mathbb{R}^n$. Since

$$A(z_{n_k}) = \frac{A(x_{n_k})}{||x_{n_k}||} = \frac{y_{n_k}}{||x_{n_k}||}$$

and $\{y_{n_k}\}$ converges to y while $\{x_{n_k}\}$ diverges, $A(z) = \lim_{n_k \to \infty} A(z_{n_k}) = 0$. We first prove that there exists a $N \in \mathbb{N}$ such that $x_{n_k} - z \in I_{n_k}$ for all $k \geq N$. Then, we prove that there exists a $P \geq N$ for which $||x_P - z|| < ||x_P||$. This will contradict our assumption that x_P is a minimal normed vector of I_P.

Claim 2.4.1. There exists a $N \in \mathbb{N}$ such that $x_{n_k} - z \in I_{n_k}$ for all $k \geq N$.

Proof. For any $v \in \mathbb{R}^p$, we have, v_i, for $1 \leq i \leq p$, denote the i^{th} coordinate of v. Let $J = \{i \mid z_i > 0 \text{ where } 1 \leq i \leq p\}$. For all $i \notin J$,

$$(x_n - z)_i = (x_n)_i - z_i \geq (x_n)_i \geq 0$$

Let $i \in J$. For $n \geq M_0$, $x_n = ||x_n|| \cdot (z_n)$. Thus, $(x_n)_i = ||x_n|| \cdot (z_n)_i$. Since $z_i > 0$, there is a $N_0 \in \mathbb{N}$ such that for all $n \geq N_0$, $(z_n)_i > 0$. Since $||x_n||$ blows up to infinity, there exists a $N_i \geq M_0$ such that $(x_{n_k})_i - (z_{n_k})_i > z_i$ for all $k \geq N_i$. Also, since $(z_{n_k})_i \to z_i$, there is a $M_i \geq M_0$ such that $|(z_{n_k})_i - z_i| < \frac{z_i}{2}$ for all $k \geq M_i$.

Let $N = \max\{N_1, N_2 \ldots N_p, M_1, M_2 \ldots M_p\}$. Then, for all $k \geq N$, for all $i \in J$,

$$(x_{n_k})_i - z_i > \frac{z_i}{2} > 0$$

This implies that $x_{n_k} - z \in X_p$ for all such k. Since $A(z) = 0$, $A(x_{n_k} - z) = y_{n_k}$. Thus, $x_{n_k} - z \in I_{n_k}$ for all $k \geq N$. ◇

Claim 2.4.2. $||x_P - z|| < ||x_P||$ for some $P \geq N$, $P \in \mathbb{N}$.

Proof. We know, if θ_n is the angle between the vectors x_n and z,

$$\begin{aligned} ||x_n - z||^2 &= ||x_n||^2 + ||z||^2 - 2\langle x_n, z\rangle \\ &= ||x_n||^2 + 1 - 2||x_n|| \cdot ||z||\theta_n \\ &= ||x_n||^2 + 1 - 2||x_n||\theta_n \end{aligned} \tag{2.1}$$

Since z_n is defined to be $\frac{x_n}{||x_n||}$ for all $n \geq M_0$, θ_n is also the angle between z_n and z. Define $f\colon \mathbb{R}^p \mapsto \mathbb{R}$ by $f(v) = \langle v, z\rangle = ||v|| \cdot \cos(\theta)$ where θ is the

angle between v and z. Clearly, f is continuous. Since $\{z_{n_k}\} \to z$, $\{f(z_{n_k})\}$ converges to $||z||^2 = 1$. Also, $f(z_{n_k}) = \cos(\theta_{n_k})$. This implies that $\{\cos(\theta_{n_k})\}$ converges to 1. Thus, for some $P \geq N$, $\cos(\theta_P) > \frac{1}{2}$. Substituting for $\cos(\theta_P)$ in Equation (2.1), we have $||x_P - z||^2 < ||x_P||^2$. $\diamond$

Thus, $x_P - z \in I_P$ and $||x_P - z|| < ||x_P||$. This contradicts our assumption that $||x_P|| = \inf_{x \in I_P} ||x||$. This implies that our supposition requiring $\{x_n\}$ to be unbounded is incorrect. This completes the proof of the theorem. $\square$

Theorem 2.4 can be extended to normed linear spaces over $\mathbb{R}$. This is done in the Appendix of the report.

Lemma 2.5 (Farkas' Lemma I). Let $A \in \mathcal{M}(n, m)$ and $b \in \mathbb{R}^n$. Then, exactly one of these two linear programs have a solution:

$$\begin{aligned} Ax &= b \\ x &\in X_m \end{aligned} \qquad (2.2) \qquad\qquad \begin{aligned} \langle b, y \rangle &< 0 \\ A^T y &\in X_m \\ y &\in \mathbb{R}^n \end{aligned} \qquad (2.3)$$

Proof. First, we prove that both the systems cannot be feasible at the same time. Assume otherwise. Then, there exists and $x \in X_m$ and $y \in \mathbb{R}^n$ such that

$$0 > \langle b, y \rangle = \langle Ax, y \rangle = \langle x, A^T y \rangle \geq 0$$

Thus, our supposition is false. Now, assume that System 2.2 is infeasible. We prove that System 2.3 is feasible. Since System 2.2 is infeasible, $b \notin A(X_m)$. By Theorem 2.4, $A(X_m)$ is closed. Clearly, it is also convex and non-empty. By the *Hyperplane Separation Lemma*, there exists $a \in \mathbb{R}^n$ and $\gamma \in \mathbb{R}$ such that $\langle Ax, a \rangle = \langle x, A^T a \rangle \geq \gamma$ for all $x \in X_m$ and $\langle b, a \rangle < \gamma$. Since $0 \in A(X_m)$, γ is non-positive. This implies that $\langle b, a \rangle < 0$. We complete the proof by showing that $(A^T a)_j \geq 0$ for all $1 \leq j \leq m$. Assume that there is an index p such that $(A^T a)_p < 0$. Let $\{e_j\}_{j=1}^m$ be the usual basis of $\mathbb{R}^m$. Consider the following $x \in \mathbb{R}^m$,

$$x = \begin{cases} e_p & \text{if } \gamma = 0 \\ \frac{\gamma}{2(A^T y)_p} e_p, & \text{if } \gamma < 0 \end{cases}$$

Note that $x \in X_m$ in both these cases. Clearly, $\langle x, A^T a \rangle < \gamma$ which contradicts the inference we made using the *Hyperplane Separation Lemma*. $\square$

Lemma 2.6 (Farkas' Lemma II). Let $A \in \mathcal{M}(n,m)$ and $b \in \mathbb{R}^n$. Then, exactly one of these two linear programs have a solution:

$$\begin{aligned} Ax + s &= b \qquad (2.4) \\ x &\in X_m \\ s &\in X_n \end{aligned} \qquad\qquad \begin{aligned} \langle b, y\rangle &< 0 \qquad (2.5) \\ A^T y &\in X_m \\ y &\in X_n \end{aligned}$$

Proof. Define $A_0 \in \mathcal{M}(n, m+n)$ as follows:

$$A_0 = \left(\begin{array}{c|c} A_{n,m} & I_{n,n} \end{array} \right)$$

Then, for $y \in \mathbb{R}^n$,

$$A_0^T y \in X_{n+m} \implies \left(\frac{A_{m,n}^T}{I_{n,n}} \right) y \in X_{n+m} \implies A^T y \in X_m \text{ and } y \in X_n$$

Thus, the following two linear programs are equivalent to the ones we are given:

$$\begin{aligned} A_0 \cdot (x,s)^T &= b \qquad (2.6) \\ (x,s)^T &\in X_{n+m} \end{aligned} \qquad\qquad \begin{aligned} \langle b, y\rangle &< 0 \qquad (2.7) \\ A_0^T y &\in X_{n+m} \end{aligned}$$

On applying Lemma 2.5 to the systems above, the result follows. □

Theorem 2.7 (Strong Duality). Let x^* be an optimal solution of the primal LP (System 2.8) and y^* be an optimal solution of the dual LP (System 2.9). Then, $\langle b, y^*\rangle = \langle c, x^*\rangle$.

$$\begin{aligned} &\text{Maximize } \langle c, x\rangle \text{ subject to} \qquad (2.8) \\ &(Ax)_i \leq b_i \text{ for all } 1 \leq i \leq n \\ &x \in X_m \end{aligned} \qquad\qquad \begin{aligned} &\text{Minimize } \langle b, y\rangle \text{ subject to} \qquad (2.9) \\ &(A^T y)_j \geq c_j \text{ for all } 1 \leq j \leq m \\ &y \in X_n \end{aligned}$$

Proof. By the *Weak Duality Theorem*, we know that $\langle b, y^*\rangle \geq \langle c, x^*\rangle$. We prove that for any $\alpha \in \mathbb{R}$ such that $\langle c, x^*\rangle < \alpha$, $\langle b, y^*\rangle < \alpha$. This will prove that $\langle b, y^*\rangle \leq \langle c, x^*\rangle$ and thus complete the proof of the theorem. Let $\alpha \in \mathbb{R}$ such that $\langle c, x^*\rangle < \alpha$. Then, for any feasible solution x of System 2.8, $\langle c, x\rangle < \alpha$. Thus, $\langle -c, x\rangle > -\alpha$. Consider the following two linear programs.

$$\begin{aligned} (Ax)_i &\leq b_i \text{ for all } 1 \leq i \leq n \quad (2.10) \\ \langle -c, x\rangle &\leq -\alpha \\ x &\in X_m \end{aligned} \qquad \begin{aligned} \langle b, y\rangle - \alpha z &< 0 \quad (2.11) \\ A^T y - cz &\in X_m \\ y &\in X_n \\ z &\in \mathbb{R}_+ \end{aligned}$$

By our previous observation, System 2.10 is infeasible. We introduce slack variables $s \in X_n$ and $t \in \mathbb{R}_+$ so that these programs can we rewritten as equalities.

$$\begin{aligned} Ax + s &= b \quad (2.12) \\ \langle -c, x\rangle + t &= -\alpha \\ x &\in X_m \\ s &\in X_n \\ t &\in \mathbb{R}_+ \end{aligned} \qquad \begin{aligned} \langle b, y\rangle - \alpha z &< 0 \quad (2.13) \\ A^T y - cz &\in X_n \\ y &\in X_m \\ z &\in \mathbb{R}_+ \end{aligned}$$

Define $A_0 \in \mathcal{M}(m+1, m+n+1)$ as follows:

$$A_0 = \left(\begin{array}{c|c|c} A_{m,n} & I_{m,m} & 0_{m,1} \\ \hline -c_{1,n}^T & 0_{1,m} & 1_{1,1} \end{array}\right)$$

Then,

$$A_0^T = \left(\begin{array}{c|c} A_{n,m}^T & -c_{n,1} \\ \hline I_{m,m} & 0_{m,1} \\ \hline 0_{1,m} & 1_{1,1} \end{array}\right)$$

Now,

$$\begin{aligned} A_0^T \begin{pmatrix} y \\ z \end{pmatrix} \in X_{m+n+1} &\implies \left(\begin{array}{c|c} A_{n,m}^T & -c_{n,1} \\ \hline I_{m,m} & 0_{m,1} \\ \hline 0_{1,m} & 1_{1,1} \end{array}\right) \begin{pmatrix} y \\ z \end{pmatrix} \in X_{m+n+1} \\ &\implies A^T y - cz \in X_n,\ y \in X_m,\ z \in \mathbb{R}_+ \end{aligned}$$

This implies that we can further transform our linear programs as follows:

$$\begin{aligned} A_0 \cdot (x, s, t)^T &= (b, -\alpha)^T \quad (2.14) \\ (x, s, t)^T &\in X_{m+n+1} \end{aligned} \qquad \begin{aligned} \langle (b, -\alpha), (y, z)\rangle &< 0 \quad (2.15) \\ A_0^T \cdot (y, z)^T &\in X_{m+n+1} \end{aligned}$$

Since System 2.10 is infeasible, System 2.14 is infeasible as well. Applying *Farkas' Lemma I* on these two systems, we get that 2.15 is feasible. Let (y', z') be a solution of System 2.15. We consider two cases depending on the value of z'.

Case 2.7.1 ($z' = 0$). Then, the following LP is feasible:

$$\begin{aligned} \langle b, y \rangle &< 0 \\ A^T y &\in X_n \\ y &\in X_m \end{aligned}$$

Then, by *Farkas' Lemma II*, the following LP is infeasible:

$$\begin{aligned} Ax + s &= b \\ x &\in X_m \\ s &\in X_n \end{aligned}$$

This implies that the solution space of the following LP is empty:

$$\begin{aligned} (Ax)_i &\leq b_i \text{ for all } 1 \leq i \leq n \\ x &\in X_m \end{aligned}$$

Which implies that there exists no solution to the primal. This contradicts our assumption that x^* exists.

Case 2.7.2 ($z' > 0$). Since (y', z') is a solution to System 2.15, it is a solution of System 2.11. Multiplying (y', z') by $\frac{1}{z'}$, we obtain another solution of System 2.11: $(y_0, 1)$ where $y_0 = \frac{y'}{z'}$. Thus, by the first constraint of that system, $\langle b, y_0 \rangle < \alpha$. Since y^* minimizes $\langle b, y \rangle$, $\langle b, y^* \rangle \leq \langle b, y_0 \rangle < \alpha$.

Since this is true for an arbitrary $\alpha \in \mathbb{R}$, $\langle b, y^* \rangle \leq \langle c, x^* \rangle$. Clubbing this result with the *Weak Duality Theorem*, we get that $\langle b, y^* \rangle = \langle c, x^* \rangle$ which completes the proof of this theorem. □

3 Positive Operators on Real Vector Spaces

In this section, we let V be a real, finite-dimensional vector space and $\mathcal{L}(V)$ denote the set of linear functions from V to V. We let n be the dimension of

V. We treat a $X \in \mathcal{L}(V)$ both as an operator and as a matrix as the situation demands. It will be clear from the context if X is being treated as an operator or its corresponding matrix.

Definition. $X \in \mathcal{L}(V)$ is said to be *self-adjoint* if the adjoint of X is X itself. That is, $\langle Xv, w\rangle = \langle v, Xw\rangle$ for all $v, w \in V$.

We will use the following well known theorems of Linear Algebra. We state them without their proofs. The author refers the reader to [1] and [2] for the proofs of these theorems.

Theorem 3.1. $X \in \mathcal{L}(V)$ is a self-adjoint operator if, and only if, X is a symmetric matrix.

Theorem 3.2 (Real Spectral Theorem). Let $X \in \mathcal{L}(V)$. Then, the following statements are equivalent:

(i) X is self-adjoint.

(ii) V has an orthonormal basis consisting of the eigenvectors of X.

(iii) X has a diagonal matrix with respect to some orthonormal basis of V.

We now state the definition of a positive semidefinite operator and a positive operator.

Definition. The operator $X \in \mathcal{L}(V)$ is *positive semidefinite* if $\langle Xv, v\rangle \geq 0$ for all $v \in V$.

Definition. An operator on V is *positive* if it is both self-adjoint and positive semidefinite.

Before we prove the theorem which describes the equivalent conditions for an operator to be positive, we define the notion of a square root of an operator.

Definition. An operator R on V is called a *square root* of $X \in \mathcal{L}(V)$ if $R^2 = X$.

Theorem 3.3. Let $X \in \mathcal{L}(V)$. Then, the following statements are equivalent:

(i) X is positive.

(ii) X is self-adjoint and all the eigenvalues of X are non-negative.

(iii) X has a positive square root.

(iv) X has a self-adjoint square root.

(v) There exists $R \in \mathcal{L}(V)$ such that $X = R^T R$.

(vi) $X = \sum_{i=1}^{n} \lambda_i w_i w_i^T$ where $\lambda_i \geq 0$ for all i and $\{w_i\}_{i=1}^{n}$ is an orthonormal set in $\mathbb{R}^n$ (w_is are column vectors).

Proof. We prove that (i) through (v) are equivalent by proving that (i) $\Rightarrow$ (ii) $\Rightarrow$ (iii) $\Rightarrow$ (iv) $\Rightarrow$ (v) $\Rightarrow$ (i). Then, we separately show that (i) and (vi) are equivalent.

Let X be a positive operator. Then, by definition, X is self-adjoint. Let λ be an eigenvalue of X. Then, there exists a non-zero $v \in V$ such that $X(v) = \lambda v$. Since X is positive semidefinite,

$$\langle Xv, v\rangle \geq 0 \implies \langle \lambda v, v\rangle \geq 0 \implies \lambda\langle v, v\rangle \geq 0 \implies \lambda \geq 0$$

This proves that (i) $\Rightarrow$ (ii).

Now, assume (ii). Since X is self-adjoint, by part (ii) of Theorem 3.2, V has an orthonormal basis consisting of the eigenvectors of X. Let $\{w_i\}_{i=1}^{n}$ be the orthonormal set of eigenvectors of X (this is unique up to the ordering of the set). Let $\{\lambda_i\}_{i=1}^{n}$ be the set of eigenvectors of X corresponding to the w_is. Since the eigenvalues are non-negative, $\sqrt{\lambda_i}$ exists in $\mathbb{R}$. Define $R \in \mathcal{L}(V)$ by $R(w_i) = \sqrt{\lambda_i} w_i$. Clearly, $R^2(w_i) = X(w_i)$ for all $1 \leq i \leq n$. Thus, R is a square root of X. Let $v \in V$. Then, $v = \sum_{i=1}^{n} \alpha_i w_i$ where $\alpha_i \in \mathbb{R}$ for all $1 \leq i \leq n$. Then, if $\delta_{i,j}$ represents the Kronecker delta,

$$\begin{aligned}
\langle R(v), v\rangle &= \langle R(\sum_{i=1}^{n} \alpha_i w_i), \sum_{j=1}^{n} \alpha_j w_j\rangle \\
&= \sum_{i,j=1}^{n} \alpha_i \alpha_j \langle R(w_i), w_j\rangle \\
&= \sum_{i,j=1}^{n} \alpha_i \alpha_j \sqrt{\lambda_i}\langle w_i, w_j\rangle \\
&= \sum_{i,j=1}^{n} \delta_{i,j} \alpha_i \alpha_j \sqrt{\lambda_i} \\
&= \sum_{i=1}^{n} \alpha_i^2 \sqrt{\lambda_i} \\
&\geq 0
\end{aligned}$$

This proves that R is positive semidefinite. It is also clear that R is self-adjoint. Thus, R is a positive square root of X. This proves (iii).

(iii) $\Rightarrow$ (iv) follows directly as does (iv) $\Rightarrow$ (v). We now prove that (v) $\Rightarrow$ (i). Let $X = R^T R$ for some $R \in \mathcal{L}(V)$. Then, for any $v \in V$,

$$\langle X(v), v\rangle = \langle R^T R(v), v\rangle = \langle R(v), R(v)\rangle \geq 0$$

Thus, X is positive semidefinite. Clearly, $X^T = R^T R = X$. Thus, X is self-adjoint. This proves (i).

Finally, we prove that (i) $\iff$ (vi). Assume that X is positive. Then, X is self-adjoint. By part (ii) of Theorem 3.2, V has an orthonormal basis consisting of the eigenvectors of X. Let $\{w_i\}_{i=1}^n$ be the orthonormal set of eigenvectors of X. Let W be the matrix whose columns, in order, are w_i. By part (iii) of Theorem 3.2, $X = WDW^{-1}$ where D is a diagonal matrix with diagonal entries $\{\lambda_i\}_{i=1}^n$ where λ_i is the eigenvalue corresponding to the eigenvector w_i. By part (ii) of this theorem, $\lambda_i \geq 0$ for all $1 \leq i \leq n$. Since $\{w_i\}_{i=1}^n$ is an orthonormal set, $W^T W = WW^T = I$. Thus, $W^T = W^{-1}$. Thus, $X = WDW^T$. Then,

$$X = WDW^T = W \cdot (\sum_{i=1}^n \lambda_i e_i e_i^T) \cdot W^T = \sum_{i=1}^n \lambda_i W \cdot (e_i e_i^T) \cdot W^T = \sum_{i=1}^n \lambda_i w_i w_i^T$$

This proves (vi).

Now, assume that (vi) is true. That is, $X = \sum_{i=1}^n \lambda_i w_i w_i^T$ where $\lambda_i \geq 0$ for all i and $\{w_i\}_{i=1}^n$ is an orthonormal set in $\mathbb{R}^n$. Then, clearly, $X^T = X = \sum_{i=1}^n \lambda_i w_i w_i^T$. Thus, X is self-adjoint. Let $v \in V$. Then,

$$\langle Xv, v\rangle = \langle \sum_{i=1}^n \lambda_i w_i w_i^T v, v\rangle = \sum_{i=1}^n \lambda_i \langle w_i w_i^T v, v\rangle = \sum_{i=1}^n \lambda_i \langle w_i^T v, w_i^T v\rangle \geq 0$$

The last inequality follows since $\lambda_i \geq 0$ for all $1 \leq i \leq n$. Thus, X is positive semidefinite. This implies that X is a positive operator which proves (i). $\square$

Theorem 3.4. A positive operator on V has a unique positive square root.

Proof. Let $X \in \mathcal{L}(V)$ be a positive operator and v be an eigenvalue of X (such a v exists by Theorem 3.2). Let λ be the eigenvalue of X corresponding to v. Then, by Theorem 3.3, $\lambda \geq 0$. By the same theorem, we also know that there

exists a positive square root, say R, of X. We prove that $Rv = \sqrt{\lambda}v$. This will indeed prove that R is unique since the eigenvalues of X form a basis of V by Theorem 3.2.

Since R is a positive operator, by Theorem 3.2, the eigenvectors of R, say $\{u_i\}_{i=1}^n$, form an orthonormal basis of V. Furthermore, its eigenvalues are non-negative. Thus, there exists $\{\lambda_i\}_{i=1}^n$, a set of non-negative reals, such that $Ru_i = \sqrt{\lambda_i}u_i$ for all $1 \leq i \leq n$. Since the u_is form a basis of V, we can write $v = \sum_{i=1}^n \alpha_i u_i$ for some $\alpha_i \in \mathbb{R}$ for all $1 \leq i \leq n$. Thus,

$$\begin{aligned}
Rv = \sum_{i=1}^n \alpha_i \sqrt{\lambda_i} u_i &\implies R^2 v = \sum_{i=1}^n \alpha_i \lambda_i u_i \\
&\implies Xv = \sum_{i=1}^n \alpha_i \lambda_i u_i \\
&\implies \lambda v = \sum_{i=1}^n \alpha_i \lambda_i u_i \\
&\implies \sum_{i=1}^n \alpha_i \lambda u_i = \sum_{i=1}^n \alpha_i \lambda_i u_i \\
&\implies \sum_{i=1}^n \alpha_i (\lambda - \lambda_i) u_i = 0 \\
&\implies \alpha_i(\lambda - \lambda_i) = 0 \text{ for all } 1 \leq i \leq n
\end{aligned}$$

Let $I = \{i \mid 1 \leq i \leq n \text{ and } \alpha_i \neq 0\}$. Then, for all $i \in I$, $\lambda_i = \lambda$. Also, $v = \sum_{i=1}^n \alpha_i u_i = \sum_{i \in I} \alpha_i u_i$. Thus,

$$Rv = \sum_{i \in I} \alpha_i \sqrt{\lambda_i} u_i = \sum_{i \in I} \alpha_i \sqrt{\lambda} u_i = \sqrt{\lambda} \sum_{i \in I} \alpha_i u_i = \sqrt{\lambda} v$$

This completes the proof of this theorem. □

4 Semidefinite Programming

A *semidefinite program*, abbreviated as SDP, is similar to a LP in that we are required to maximize or minimize a linear function subject to some linear constraints. In addition, the square matrix corresponding to the variables is positive. Formally, given m and $n \in \mathbb{N}$; b_k, $a_{i,j}^{(k)}$, $c_{i,j} \in \mathbb{R}$ for all $1 \leq i, j \leq n$,

$1 \leq k \leq m$;

$$\begin{aligned} &\text{Maximize or minimize } \sum_{i,j=1}^{n} c_{i,j} x_{i,j} \text{ subject to} \\ &\sum_{i,j=1}^{n} a_{i,j}^{(k)} x_{i,j} = b_k \text{ for all } 1 \leq k \leq m \\ &x_{i,j} \in \mathbb{R} \text{ for all } 1 \leq i, j \leq n \\ &X = (x_{i,j}) \text{ is a positive operator} \end{aligned} \tag{4.1}$$

It is known that SDPs can be solved, up to an additive error of ϵ, in time polynomial in the size of the input and $\log(\frac{1}{\epsilon})$ [8].

Semidefinite programming is often used in the form of *vector programming* where the variables are elements in $\mathbb{R}^n$, for some $n \in \mathbb{N}$, and the objective function and the constraints are linear in the inner product of these vectors. Formally, given m and $n \in \mathbb{N}$; b_k, $a_{i,j}^{(k)}$, $c_{i,j} \in \mathbb{R}$ for all $1 \leq i, j \leq n$, $1 \leq k \leq m$; we are required to

$$\begin{aligned} &\text{Maximize or minimize } \sum_{i,j=1}^{n} c_{i,j} \langle v_i, v_j \rangle \text{ subject to} \\ &\sum_{i,j=1}^{n} a_{i,j}^{(k)} \langle v_i, v_j \rangle = b_k \text{ for all } 1 \leq k \leq m \\ &v_i \in \mathbb{R}^n \text{ for all } 1 \leq i \leq n \end{aligned} \tag{4.2}$$

Lemma 4.1. Systems 4.1 and 4.2 are equivalent.

Proof. Let X be a solution of System 4.1. Then, since X is positive, by Theorem 3.3, it has a positive square root. That is, $X = V^2 = V^T V$ for some positive operator V. Let $\{v_i\}_{i=1}^n$ be the columns, in order, of V. Then, by construction, $\langle v_i, v_j \rangle = x_{i,j}$. Thus, $\{v_i\}_{i=1}^n$ is a feasible solution of System 4.2.

Now, consider $\{v_i\}_{i=1}^n$, a feasible solution of System 4.2. Define $X = (x_{i,j})$ where $x_{i,j} = \langle v_i, v_j \rangle$. Then, if V is the matrix whose columns are $\{v_i\}_{i=1}^n$, $X = V^T V$. Thus, by Theorem 3.3, X is a positive operator. This implies that System 4.1 and System 4.2 are equivalent. □

In the rest of this section, we will look at the MAX-CUT problem and present

two approximation algorithms for it. We will first describe a simple, randomized $\frac{1}{2}$-approximation algorithm for the problem and then use semidefinite programming to design a 0.878-approximation algorithm for it. This algorithm was presented by Vandenberghe and Boyd in their seminal work on using semidefinite programming to solve optimization problems [5].

Problem. Let $G(V, E)$ be an undirected graph where $V = \{v_1, v_2 \dots v_n\}$ and $w \colon E \mapsto \mathbb{R}_+$ be a weight function defined on G. For a $e = (v_i, v_j) \in E$, we use w_e or $w_{i,j}$ to denote $w(e)$. A subset U of V is called a *cut* of G. We define $\textsc{Cut}(U) = \{e = (v_i, v_j) \in E \mid v_i \in U \text{ and } v_j \notin U\}$ and the *weight* of U to be $\sum_{e \in \textsc{Cut}(U)} w_e$. Find a cut U' of G which has the maximum weight.

Algorithm 4.2. Given a graph $G(V, E)$ where $V = \{v_1, v_2 \dots v_n\}$, place a vertex $v_i \in V$ into a set U independently with probability $\frac{1}{2}$. Report U. △

Lemma 4.3. Algorithm 4.2, after derandomization, is a $\frac{1}{2}$-approximation algorithm for the Max-Cut problem.

Proof. Define X_e to be the random variable that takes 1 if $e \in \textsc{Cut}(U)$ and 0 otherwise. Define $Y = \sum_{e \in E} w_e X_e$. Clearly, Y is the random variable that takes the value of the weight of the cut U. Now,

$$\mathbf{E}[Y] = \mathbf{E}[\sum_{e \in E} w_e X_e] = \sum_{e \in E} w_e \mathbf{E}[X_e] = \sum_{e \in E} w_e \mathbf{Pr}[e \in \textsc{Cut}(U)] \tag{4.3}$$

For $e = (v_i, v_j) \in E$, the probability that $e \in \textsc{Cut}(U)$ is the probability that exactly one of v_i and v_j are in U. Thus, $\mathbf{Pr}[e \in \textsc{Cut}(U)] = \frac{1}{2}$. Substituting for this in Equation (4.3), we get that

$$\mathbf{E}[Y] = \frac{1}{2} \sum_{e \in E} w_e \geq \frac{1}{2} \textsc{opt}$$

where $\textsc{opt}$ is the optimal value of this Max-Cut instance. Thus, after derandomization, Algorithm 4.2 produces a $\frac{1}{2}$-approximate solution for the Max-Cut problem. □

Now we move onto describing a SDP based algorithm to solve Max-Cut. First, we claim that System 4.4 models the Max-Cut problem. Then, we give a SDP relaxation of System 4.4 in System 4.5. We will prove that this relaxation admits an algorithm (Algorithm 4.6) which can be solved in polynomial time and prove

that this algorithm has an approximation factor of 0.878 in Theorem 4.7.

Lemma 4.4. The following system models the MAX-CUT problem.

$$\text{Maximize } \frac{1}{2} \sum_{(v_i,v_j)\in E} w_{i,j}(1 - y_i y_j) \text{ subject to} \tag{4.4}$$
$$y_i \in \{-1, 1\} \text{ for all } 1 \leq i \leq n$$

Proof. Consider $U \subseteq V$, a cut of the graph. Let

$$y_i = \begin{cases} 1 & \text{if } v_i \in U \\ -1 & \text{if } v_i \notin U \end{cases}$$

Clearly, this is a feasible solution of System 4.4. Similarly, given a solution of 4.4, we let U be the set of all vertices v_i such that $y_i = 1$. This is a cut. Now,

$$\begin{aligned} \frac{1}{2} \sum_{(v_i,v_j)\in E} w_{i,j}(1 - y_i y_j) &= \frac{1}{2} \sum_{(i,j)|y_i y_j = -1} w_{i,j}(1 - y_i y_j) \\ &= \sum_{(v_i,v_j)\in \text{CUT}(U)} w_{i,j} \\ &= \sum_{(v_i,v_j)\in \text{CUT}(U)} w_{i,j} \end{aligned}$$

Thus, the objective function of 4.4 maximizes the weight of the cut U. It now follows that the MAX-CUT problem is equivalent to System 4.4. □

Lemma 4.5. The following vector program is a relaxation of System 4.4.

$$\text{Maximize } \frac{1}{2} \sum_{(v_i,v_j)\in E} w_{i,j}(1 - \langle u_i, u_j \rangle) \text{ subject to} \tag{4.5}$$
$$u_i \in \mathbb{R}^n \text{ for all } 1 \leq i \leq n$$
$$||u_i|| = 1 \text{ for all } 1 \leq i \leq n$$

Furthermore, $X = (x_{i,j})$ where $x_{i,j} = \langle u_i, u_j \rangle$ is a positive operator.

Proof. Let $\{y_i\}_{i=1}^n$ be a feasible solution of System 4.4 whose input is the same graph $G(V, E)$. Let $u_i = (y_i, 0, 0 \ldots 0) \in \mathbb{R}^n$ for all $1 \leq i \leq n$. Clearly, $\{u_i\}_{i=1}^n$ is a feasible solution of 4.5. Furthermore, the objective function of 4.5 is equal to that of 4.4. Thus, System 4.5 is a relaxation of System 4.4.

Consider $X = (x_{i,j})$ where $x_{i,j} = \langle u_i, u_j \rangle$. If V is the matrix whose columns are $\{u_i\}_{i=1}^n$, $X = V^T V$. By Theorem 3.3, X is a positive operator. □

These two lemmas imply that the optimal value of System 4.5, say OPT′, can be found in polynomial time (up to a small error) and that OPT′ ≥ OPT where OPT is the optimal solution of System 4.4. We now present a randomized procedure for the MAX-CUT problem. We then prove that this is a 0.878-approximation algorithm in Theorem 4.7.

Algorithm 4.6. Let $\{u_i\}_{i=1}^n$ be an optimal solution to the SDP formulation (System 4.5) of the MAX-CUT problem. Pick a vector $r = (r_1, r_2 \dots r_n) \in \mathbb{R}^n$ randomly by picking each component independently from $\mathcal{N}(0, 1)$, the normal distribution with mean 0 and variance 1. Report the cut U of G where U is constructed as follows: $v_i \in U$ if, and only if, $\langle u_i, r \rangle \geq 0$.

Theorem 4.7. Algorithm 4.6 (after derandomization), is a 0.878-approximation algorithm for the MAX-CUT problem that runs in polynomial time.

Proof. Let X_e be random variable that takes 1 if the edge $e \in \text{CUT}(U)$ and is 0 otherwise. Let W be a random variable defined by

$$W = \sum_{e \in E} w_e X_e$$

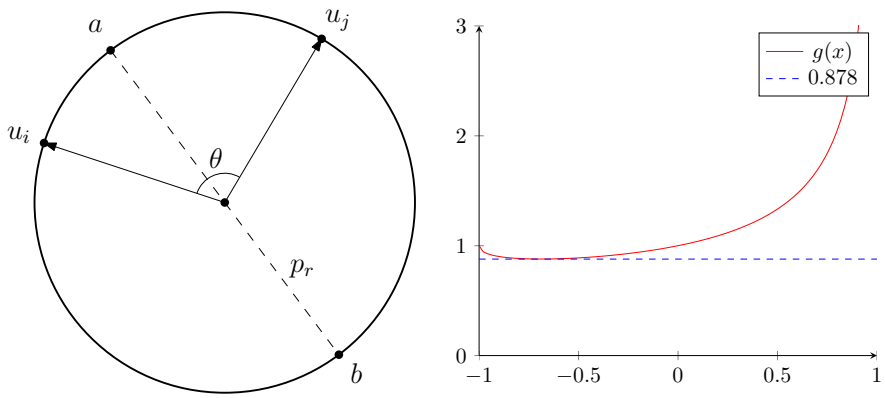
Clearly, W is the random variable that gives the weight of the cut. Then,

$$\mathbf{E}[W] = \sum_{e \in E} w_e \mathbf{E}[X_e] = \sum_{e \in E} w_e \mathbf{Pr}[e \in \text{CUT}(U)] \tag{4.6}$$

Note that the setup in Equation (4.6) and Equation (4.3) is exactly the same.

Claim 4.7.1. For $e = (v_i, v_j) \in E$, $\mathbf{Pr}[(v_i, v_j) \in \text{CUT}(U)] = \frac{1}{\pi} \cos^{-1}(\langle u_i, u_j \rangle)$.

Proof. Since $||u_i|| = ||u_j|| = 1$, $\cos^{-1}(\langle u_i, u_j \rangle) = \theta$ where θ is the angle between the vectors u_i and u_j. Let p_r be the hyperplane passing through the origin whose normal is r. Clearly, $(v_i, v_j) \in \text{CUT}(U)$ if, and only if, u_i and u_j lie on opposing sides of the unit sphere with respect to p_r. There exists a circle whose center is the origin and it passes through both u_i and u_j. The hyperplane p_r cuts this circle at two diametrically opposite points, say at a and b. Thus, $(v_i, v_j) \in \text{CUT}(U)$ if, and only if, u_i and u_j lie on opposing sides of the circle with respect to the line segment $\overline{ab}$. This is shown in Figure 2a.



(a) (b)

Figure 2: (a) illustrates the angle between the vectors u_i and u_j on a planar section of the unit sphere described in the proof of Claim 4.7.1. The plane that whose normal is r is denoted by p_r and its intersection with the circle are marked by a and b. In (b), the graph of the ratio of $\frac{1}{\pi}\cos^{-1}(x)$ and $\frac{1}{2}(1-x)$ is illustrated.

If $Y_k \sim \mathcal{N}(0,1)$ are independent random variables for all $1 \leq k \leq n$, $Y \sim \prod_{k=1}^{n} Y_k$ has the probability distribution function

$$f(y) = \prod_{k=1}^{n} \frac{1}{\sqrt{2\pi}} \cdot \mathrm{e}^{-\frac{y_k^2}{2}} = \frac{1}{(2\pi)^{\frac{n}{2}}} \cdot \mathrm{e}^{-\frac{||y||^2}{2}}$$

This implies that picking r, and thus p_r, was a spherically symmetric process. Thus, $\mathbf{Pr}[(v_i, v_j) \in \textsc{Cut}(U)]$ is the probability that only one of a or b lie within the arc defined by v_i and v_j. That is,

$$\mathbf{Pr}[(v_i, v_j) \in \textsc{Cut}(U)] = 2 \cdot \frac{\theta}{2\pi} = \frac{1}{\pi} \cdot \cos^{-1}(\langle u_i, u_j \rangle)$$

This completes the proof of the claim. ◇

Using this in Equation (4.6), we have

$$E[W] = \sum_{(v_i,v_j)\in E} w_{i,j} \cdot \frac{1}{\pi}\cos^{-1}(\langle u_i, u_j \rangle) \tag{4.7}$$

Claim 4.7.2. For all $x \in [-1, 1]$, $\frac{1}{\pi} \cdot \cos^{-1}(x) \geq 0.878 \cdot \frac{1}{2}(1-x)$.

Proof. Computing the minima of the ratio of $\frac{1}{\pi}\cos^{-1}(x)$ and $\frac{1}{2}(1-x)$ gives us

the required result. Figure 2b is a plot of $g(x) = \frac{2\cos^{-1}(x)}{\pi(1-x)}$ for $x \in [-1, 1]$. ◇

Substituting this in Equation (4.7), we get

$$E[W] \geq 0.878 \cdot \frac{1}{2} \sum_{(v_i, v_j) \in E} w_{i,j}(1 - \langle u_i, u_j \rangle) \geq 0.878 \cdot \text{OPT}'$$

This implies that there exists a cut whose weight is at least $0.878 \cdot \text{OPT}'$. Then, $\text{OPT}' \geq \text{OPT} \geq 0.878 \cdot \text{OPT}'$ implying that this procedure is indeed a 0.878-approximation algorithm for the MAX-CUT problem. Since SDPs can be solved (up to a small error) in polynomial time and picking r can be done in polynomial time, this algorithm can be implemented in polynomial time. □

Acknowledgements

The author would like to thank Dr. Aritra Banik and Dr. Sutanu Roy for helpful discussions. He would also like to thank math.stackexchange users Prudii Arca and daw for their answers to the author's questions on the platform.

References

[1] Sheldon Axler. "Operators on Inner Product Spaces". In: *Linear Algebra Done Right*. Cham: Springer International Publishing, 2015, pp. 203–240. ISBN: 978-3-319-11080-6. URL: https://doi.org/10.1007/978-3-319-11080-6_7.

[2] Kenneth Hoffman and Ray A. Kunze. *Linear Algebra*. PHI Learning, 2004. ISBN: 8120302702. URL: http://www.worldcat.org/isbn/8120302702.

[3] S. Kesavan. "Hahn-Banach Theorems". In: *Functional Analysis*. Gurgaon: Hindustan Book Agency, 2009, pp. 69–96. ISBN: 978-93-86279-42-2. URL: https://doi.org/10.1007/978-93-86279-42-2_3.

[4] Luca Trevisan. *Lecture Notes in Optimization and Algorithmic Paradigms*. Feb. 2011. URL: https://people.eecs.berkeley.edu/~luca/cs261.

[5] Lieven Vandenberghe and Stephen P. Boyd. "Semidefinite Programming". In: *SIAM Rev.* 38.1 (1996), pp. 49–95. URL: https://doi.org/10.1137/1038003.

[6] Robert J. Vanderbei. "The Simplex Method". In: *Linear Programming: Foundations and Extensions*. Cham: Springer International Publishing, 2020, pp. 11–25. ISBN: 978-3-030-39415-8. URL: https://doi.org/10.1007/978-3-030-39415-8_2.

[7] Kevin Wayne. *Linear Programming II*. July 2017. URL: https://www.cs.princeton.edu/~wayne/teaching/.

[8] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. 1st. USA: Cambridge University Press, 2011. ISBN: 0521195276. URL: https://dl.acm.org/doi/book/10.5555/1971947.

Appendix

Here, we prove that the result obtained in Theorem 2.4 can be generalized to a result for normed linear spaces over $\mathbb{R}$. Throughout this section, we let V be a normed linear space over $\mathbb{R}$.

Definition. $C \subseteq V$ is said to be a *cone* if $0 \in C$ and for all $x \in C$, $\lambda x \in C$ for all $\lambda \geq 0$.

Theorem 4.8. Let v_i, for all $1 \leq i \leq n$, be a vector of V. Define

$$C = \{\sum_{i=1}^{n} \lambda_i v_i \mid \lambda_i \geq 0 \text{ for all } 1 \leq i \leq n\}$$

Then, C is a cone that is convex and closed.

Note that this generalizes Theorem 2.4 which states that $A(X_n)$ is closed since

$$A(X_n) = \{\sum_{i=1}^{n} \lambda_i A(e_i) \mid \lambda_i \geq 0 \text{ for all } 1 \leq i \leq n\}$$

where $\{e_i\}_{i=1}^n$ is the usual basis of $\mathbb{R}^n$.

Proof. Clearly, C is convex cone. We prove that C is indeed closed by taking two cases.

Case 4.8.1 ($\{v_i\}_{i=1}^n$ is linearly independent). Consider a sequence $\{x_m\}$ in C. Thus, $x_m = \sum_{i=1}^n \lambda_i^{(m)} v_i$ for $\lambda_i^{(m)} \geq 0$ for all $1 \leq i \leq n$ for all $m \in \mathbb{N}$. Define $W = \text{SPAN}(v_1, v_2 \dots v_n)$. Since W is a finite dimensional normed linear space, it is closed. Since $\{x_m\}$ is also a sequence of W, it converges to a point $x =$

$\sum_{i=1}^{n} \lambda_i v_i \in W$. Thus, $\lambda_i^{(m)} \to \lambda_i$ as $m \to \infty$ since $|| \sum_{i=1}^{n} (\lambda_i - \lambda_i^{(m)}) v_i || \to 0$ as $m \to \infty$. Since, for all $1 \leq i \leq n$ and $m \in \mathbb{N}$, $\lambda_i^{(m)} \geq 0$, $\lambda_i \geq 0$ for all $1 \leq i \leq n$. Thus, $x \in C$. As the sequence $\{x_m\}$ was arbitrary, C is closed.

Case 4.8.2 ($\{v_i\}_{i=1}^n$ is not linearly independent). There exists $\{\alpha_i\}_{i=1}^n$ in $\mathbb{R}^n$, not all 0, such that $\sum_{i=1}^{n} \alpha_i v_i = 0$. Multiplying by -1 if necessary, we make sure that the set

$$J = \{i \mid \alpha_i < 0 \text{ where } 1 \leq i \leq n\}$$

is non-empty. Let $v = \sum_{i=1}^{n} \lambda_i v_i \in C$ where $\lambda_i \geq 0$ for all $1 \leq i \leq n$. Since $\sum_{i=1}^{n} \alpha_i v_i = 0$, for any $t \in \mathbb{R}$, $v = \sum_{i=1}^{n} (\lambda_i + t\alpha_i) v_i$. Let

$$t = \min_{i \in J} \{-\frac{\lambda_i}{\alpha_i}\}$$

Clearly, $t \geq 0$. Also, for all $i \notin J$, $\lambda_i + t\alpha_i \geq \lambda_i \geq 0$ since $\alpha_i \geq 0$. For all $i \in J$,

$$t \leq -\frac{\lambda_i}{\alpha_i} \implies -t\alpha_i \leq \lambda_i \implies \lambda_i + t\alpha_i \geq 0$$

Furthermore, there is an index j such that $\lambda_j + t\alpha_j = 0$. Thus, $v = \sum_{i \neq j} (\lambda_i + t\alpha_i) v_i$ where $\lambda_i + t\alpha_i \geq 0$ for all indices i. This implies that $C = \cup_{j=1}^{n} C_j$ where

$$C_j = \{v = \sum_{i \neq j} \lambda_i v_i \mid \lambda_i \geq 0 \text{ for all } 1 \leq i \leq n\}$$

Since each C_j is generated by fewer elements than C, we can iterate this procedure finitely many times to write C as a finite union of conic sets generated by a linearly independent set. By the first case, each of these conic sets are closed. Thus, C is closed.

This completes the proof of this lemma. □