M2OD – Optimization and Discrete Mathematics

Dr Brad Baxter
Department of Mathematics
Imperial College, London SW7 2BZ
`b.baxter@ic.ac.uk`
`www.ma.ic.ac.uk/~baxter`

These notes contain all of the examinable theory and algorithms for the first 9 lectures of M2OD. I shall provide an example sheet in Week 2 and announce office hours later in Week 1. Please note that I am on sabbatical this term, so will not often be available on campus after Week 3.

1. Linear Programming

1.1. Introduction

A *linear programming (LP)* problem requires us to find the smallest value of a linear function subject to some linear equality and inequality constraints. Thus the most general problem of this type is to

$$\begin{aligned} \text{Minimize} \quad & f(\mathbf{x}) = \mathbf{c}^T\mathbf{x}, && \mathbf{x} \in \mathbb{R}^n \\ \text{subject to} \quad & \ell_j(\mathbf{x}) = 0, && 1 \le j \le M, \\ \text{and} \quad & \ell_j(\mathbf{x}) \ge 0, && M+1 \le j \le M+P, \end{aligned} \tag{1.1}$$

where $\ell_j(\mathbf{x}) = \mathbf{a}_j^T\mathbf{x} - b_j$, for $j = 1, 2, \ldots, M+P$. However, it will suffice to focus on a more restricted problem. The *standard form* we shall consider is as follows.

$$\begin{aligned} \text{Minimize} \quad & f(\mathbf{x}) = \mathbf{c}^T\mathbf{x}, \qquad \mathbf{x} \in \mathbb{R}^n \\ \text{subject to} \quad & A\mathbf{x} = \mathbf{b} \quad \text{and} \quad \mathbf{x} \ge 0. \end{aligned} \tag{1.2}$$

Here the notation "$\mathbf{x} \ge 0$" simply means that every component of the vector $\mathbf{x}$ is non-negative.

We shall always assume that the matrix A is $m \times n$, where $m < n$, and that its columns $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$ generate a subspace of dimension m. In other words, we assume rank $A = m$. Thus the set of solutions

$$\Sigma(\mathbf{b}) \equiv \{\mathbf{x} \in \mathbb{R}^n : A\mathbf{x} = \mathbf{b}\} \tag{1.3}$$

is nonempty. Further, if $\mathbf{z} \in \Sigma(\mathbf{b})$, then

$$\Sigma(\mathbf{b}) = \mathbf{z} + K, \qquad \text{where} \qquad K = \{\mathbf{y} \in \mathbb{R}^n : A\mathbf{y} = 0\}, \tag{1.4}$$

because, if $\mathbf{x}, \mathbf{y} \in \Sigma(\mathbf{b})$, then $A(\mathbf{x} - \mathbf{y}) = \mathbf{b} - \mathbf{b} = 0$.

The constraints in (1.2) require that $\mathbf{x} \in \Sigma(\mathbf{b})$ be a vector with non-negative components. We shall use some special terminology to describe this set.

Definition 1.1. (i) If $\mathbf{x} \in \Sigma(\mathbf{b})$, then we define $B(\mathbf{x}) = \{k \in [1, n] : x_k \neq 0\}$. We shall call $B(\mathbf{x})$ the *basic set.* The nonzero variables $\{x_k : k \in B(\mathbf{x})\}$ are called *basic variables.*

(ii) If $\mathbf{x} \in \Sigma(\mathbf{b})$ and the columns of A correponding to the basic variables $\{\mathbf{a}_k : k \in B(\mathbf{x})\}$ are linearly independent, then we say that $\mathbf{x}$ is a *basic point* in $\Sigma(\mathbf{b})$. Further, if $|B(\mathbf{x})| < m$, then we say that $\mathbf{x}$ is a *degenerate basic point.*

Exercise 1.1. If $\mathbf{x} \in \Sigma(\mathbf{b})$ is a basic point, then $|B(\mathbf{x})| \leq m$.

Definition 1.2. The *feasible set* is defined by

$$S = \{\mathbf{x} \in \Sigma(\mathbf{b}) : \mathbf{x} \geq 0\}. \tag{1.5}$$

An element of S is called a *feasible point.* If $\mathbf{x} \in S$ is a basic point in $\Sigma(\mathbf{b})$, then we say that $\mathbf{x}$ is a *vertex*, also called a *basic feasible point.* We shall elucidate the geometric name in due course. Further, S is sometimes called a *simplex.*

Thus the standard LP problem becomes

$$\begin{aligned} \text{Minimize} \qquad & f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} \\ \text{subject to} \qquad & \mathbf{x} \in S. \end{aligned} \tag{1.6}$$

Definition 1.3. Any solution to (1.6) is called an *optimal feasible point.* In other words, $\mathbf{x}^* \in S$ is an optimal feasible point if and only if

$$\mathbf{c}^T\mathbf{x}^* \leq \mathbf{c}^T\mathbf{x}, \qquad \text{for all } \mathbf{x} \in S.$$

It is easy to construct LP problems for which there are *no* optimal feasible points, or infinitely many.

Example 1.1. (i) For the three-dimensional LP problem

$$\begin{aligned} \text{Minimize} \quad &: \quad x \\ \text{subject to} \quad &: \quad x + y + z = 1 \quad \text{and} \quad x \geq 0, y \geq 0, z \geq 0, \end{aligned}$$

the set of optimal feasible points is

$$\{(0, y, z) \in \mathbb{R}^3 : y + z = 1, y \geq 0, z \geq 0\}.$$

If you sketch this, you will obtain a triangle for which one edge consists of optimal feasible points.

(ii) The one-dimensional LP problem

$$\text{Minimize } -x \text{ subject to } x \geq 0$$

has no optimal feasible points.

(iii) The two-dimensional LP problem

$$\text{Minimize } -x, \text{ subject to } x \geq 0 \text{ and } 0 \leq y \leq 2,$$

has no optimal feasible point.

The following lemma might seem rather unimportant at first sight, but is crucial to the theory and algorithms of linear programming.

Lemma 1.1. Let $\mathbf{x} \in S$ be a feasible point that is not a vertex. Then we can construct a new feasible point $\mathbf{z} \in S$ for which $B(\mathbf{z}) \subset B(\mathbf{x})$ and $|B(\mathbf{z})| \leq |B(\mathbf{x})| - 1$.

Proof. By definition of vertex, the columns $\{\mathbf{a}_k : k \in B(\mathbf{x})\}$ corresponding to the basic set $B(\mathbf{x})$ are linearly dependent. Thus there are scalars $\{y_k : k \in B(\mathbf{x})\}$, not all of which are zero, such that

$$\sum_{k \in B(\mathbf{x})} y_k \mathbf{a}_k = 0,$$

so that we have a nonzero vector $\mathbf{y} \in \mathbb{R}^n$ such that $B(\mathbf{y}) \subset B(\mathbf{x})$ and

$$A\mathbf{y} = 0.$$

Hence

$$A(\mathbf{x} - t\mathbf{y}) = \mathbf{b}, \qquad \text{for all } t \in \mathbb{R},$$

and $\mathbf{x} - t\mathbf{y} \in \Sigma(\mathbf{b})$ for all $t \in \mathbb{R}$. However, if $y_k \neq 0$, then $x_k - ty_k < 0$ when $|t| > |x_k/y_k|$ and $\text{sign}(t) = \text{sign}(y_k)$. Of course, we see that $\mathbf{x} - t\mathbf{y}$ is feasible for all sufficiently small $|t|$. If we now choose the largest modulus t^* such that $\mathbf{z} \equiv \mathbf{x} - t^*\mathbf{y} \in S$, then $B(\mathbf{z}) \subset B(\mathbf{x})$, by construction, and $|B(\mathbf{z})| < |B(\mathbf{x})|$, as required. □

Theorem 1.2. The feasible set contains a vertex.

Proof. If $\mathbf{x} \in S$ is not a vertex, then Lemma 1.1 implies the existence of a vector $\mathbf{z}_1 \in S$ such that $B(\mathbf{z}_1) \subset B(\mathbf{x})$ and $|B(\mathbf{z}_1)| \leq |B(\mathbf{x})| - 1$. If $\mathbf{z}_1$ is not a vertex, then we can apply the lemma again to obtain $\mathbf{z}_2 \in S$ such that $B(\mathbf{z}_2) \subset B(\mathbf{x})$ and $|B(\mathbf{z}_2)| \leq |B(\mathbf{x})| - 2$. Of course, we apply the lemma repeatedly until we obtain a feasible point, $\mathbf{z}_p \in S$ say, that is a vertex. Thus we have $B(\mathbf{z}_p) \subset B(\mathbf{x})$ and $|B(\mathbf{z}_p)| \leq |B(\mathbf{x})| - p$. □

Theorem 1.3. If there exists an optimal feasible point, then there's an optimal vertex.

Proof. Let $\mathbf{x} \in S$ be an optimal feasible point that is not a vertex. Following the proof of Lemma 1.1, there is a nonzero vector $\mathbf{y} \in \mathbb{R}^n$ such that $B(\mathbf{y}) \subset B(\mathbf{x})$ and $\mathbf{x} - t\mathbf{y}$ is feasible for, say, $|t| < \delta$. If $f(\mathbf{y}) \neq 0$, then $\mathbf{x} - t\,\mathrm{sign}(f(\mathbf{y}))\mathbf{y} \neq \mathbf{x}$ is a feasible point for $0 < t < \delta$ such that

$$f(\mathbf{x} - t\,\mathrm{sign}(f(\mathbf{y}))\mathbf{y}) = f(\mathbf{x}) - t\,\mathrm{sign}(f(\mathbf{y}))f(\mathbf{y}) = f(\mathbf{x}) - t|f(\mathbf{y})| < f(\mathbf{x}),$$

which contradicts the optimality of $\mathbf{x}$. Thus we must have $f(\mathbf{y}) = 0$. Therefore each application of Lemma 1.1 results in a feasible point $\mathbf{z}$ such that $|B(\mathbf{z})| \le |B(\mathbf{x})| - 1$ and $f(\mathbf{z}) = f(\mathbf{x})$. Thus we can apply the lemma several times, as in the proof of Theorem 1.2, to obtain an optimal vertex. □

Definition 1.4. If $1 \le m \le n$, then an m-set is simply any subset of $\{1, 2, \ldots, n\}$ containing m different numbers.

Exercise 1.2. There are $\binom{n}{m}$ m-sets.

Proposition 1.4. There are at most $\binom{n}{m}$ vertices of S.

Proof. Suppose that $\mathbf{x} \in S$ is a nondegenerate vertex, so that $|B(\mathbf{x})| = m$. If $\mathbf{y} \in S$ is a vertex such that $B(\mathbf{y}) = B(\mathbf{x})$, then $\mathbf{x} = \mathbf{y}$, because the equation

$$0 = A(\mathbf{x} - \mathbf{y}) = \sum_{k \in B(\mathbf{x})} (x_k - y_k)\,\mathbf{a}_k,$$

and the linear independence of $\{\mathbf{a}_k : k \in B(\mathbf{x})\}$ imply that $\mathbf{x} = \mathbf{y}$. Thus every nondegenerate vertex corresponds to exactly one m-set of $\{1, 2, \ldots, n\}$.

If $\mathbf{x} \in S$ is degenerate, then choose any m-set B^* such that $B(\mathbf{x}) \subset B^*$ and the vectors $\{\mathbf{a}_\ell : \ell \in B^*\}$ are linearly independent. (This is possible because of the assumption rank $A = m$.) If $\mathbf{y} \in S$ is any vertex for which $B(\mathbf{y}) \subset B^*$, then we obtain

$$0 = A(\mathbf{x} - \mathbf{y}) = \sum_{k \in B^*} (x_k - y_k)\,\mathbf{a}_k,$$

and we deduce that $\mathbf{x} = \mathbf{y}$. Thus we have associated exactly one m-set of $\{1, 2, \ldots, n\}$ with each degenerate vertex.

We have therefore established that every vertex, degenerate or nondegenerate, corresponds to exactly one m-set of $\{1, 2, \ldots, n\}$. Thus the number of vertices is at most the number of m-sets, which is $\binom{n}{m}$. □

The last result suggests a simple algorithm to compute the minimum of an LP problem when an optimal feasible point exists:

(i) We first find all the vertices: For each m-set B, we solve the equations

$$\sum_{k \in B} x_k \mathbf{a}_k = \mathbf{b}$$

if the vectors $\{\mathbf{a}_k : k \in B\}$ are linearly independent. If the computed vector satisfies $\mathbf{x} \geq 0$, then we have found a vertex. Let V denote the set of all vertices — we know that $|V| \leq \binom{n}{m}$.

(ii) We now compute $f(\mathbf{v}) = \mathbf{c}^T\mathbf{v}$ for every vertex $\mathbf{v} \in V$ and pick any element giving the smallest value of f.

Unfortunately this method is only useful when $\binom{n}{m}$ is small, which excludes almost all problems of practical interest, since $\binom{n}{m}$ is *enormous* for quite modest values of m and n. However, it does suggest a useful strategy: Since the minimum value of $f(\mathbf{x})$ is attained at a vertex, then we can attempt the solve the problem

$$\begin{aligned} \text{Minimize} \quad & f(\mathbf{x}) = \mathbf{c}^T\mathbf{x}, \qquad \mathbf{x} \in \mathbb{R}^n \\ \text{subject to} \quad & \mathbf{x} \text{ is a vertex of } S, \end{aligned} \tag{1.7}$$

by starting at a vertex and then moving to a vertex at which $f(\mathbf{x})$ takes a smaller value. If we regard the cost vector $\mathbf{c}$ as defining our “up” direction, then we simply move from each vertex to a lower vertex. This is a simple version of the simplex method.

It can be shown that

$$\lim_{n\to\infty} \frac{\binom{2n}{n}}{2^{2n}/\sqrt{\pi n}} = 1.$$

Therefore the denominator provides a simple way to estimate the numerator. Choosing $n = 200$, we find that the number of vertices in a linear programming problem with 400 vertices and 200 constraints can have more vertices than there are atoms in our universe! In contrast, the simplex algorithm typically visits a small multiple of n vertices before finding the solution, so that $n = 200$ problems can be solved with ease on any basic Pentium.

1.2. Moving between Vertices

The idea of the simplex method is to move from vertex to vertex of S to reduce the value of f, ultimately locating an optimal vertex. Therefore suppose we have located a **nondegenerate vertex $\mathbf{v} \in S$**. Any other vertex must have a different basic set of variables. Thus, in moving from $\mathbf{v}$ to another vertex, one of the basic variables must become zero. To express this, we introduce the following notation.

Definition 1.5. Let $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n$ be the coordinate vectors, that is

$$\mathbf{e}_1 = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \mathbf{e}_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots, \\ 0 \end{pmatrix}, \ldots, \mathbf{e}_n = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} \in \mathbb{R}^n.$$

Further, if $j_1, j_2, \ldots, j_n$ is a permutation of the integers $1, 2, \ldots, n$, then we say that the matrix

$$\begin{pmatrix} \mathbf{e}_{j_1} & \mathbf{e}_{j_2} & \cdots & \mathbf{e}_{j_n} \end{pmatrix}$$

is a permutation matrix. In other words, we're simply shuffling the columns of the identity matrix.

Let's write

$$B \equiv B(\mathbf{v}) = \{j_1, j_2, \ldots, j_m\} \text{ and } N = \{1, 2, \ldots, n\} \backslash B \equiv \{k_1, k_2, \ldots, k_{n-m}\}. \tag{1.8}$$

Thus, for any vector $\mathbf{x} \in \mathbb{R}^n$ (not necessarily in S) we can write

$$\mathbf{x} = \sum_{\ell=1}^{m} x_{j_\ell} \mathbf{e}_{j_\ell} + \sum_{p=1}^{n-m} x_{k_p} \mathbf{e}_{k_p}. \tag{1.9}$$

Accordingly, we set

$$\mathbf{x}_B = \begin{pmatrix} x_{j_1} \\ \vdots \\ x_{j_m} \end{pmatrix} \in \mathbb{R}^m \quad \text{and} \quad \mathbf{x}_N = \begin{pmatrix} x_{k_1} \\ \vdots \\ x_{k_{n-m}} \end{pmatrix} \in \mathbb{R}^{n-m}. \tag{1.10}$$

Similarly, we define

$$\mathbf{v}_B = \begin{pmatrix} v_{j_1} \\ \vdots \\ v_{j_m} \end{pmatrix} \in \mathbb{R}^m. \tag{1.11}$$

Further, we shall write

$$A_B = \begin{pmatrix} \mathbf{a}_{j_1} & \mathbf{a}_{j_2} & \cdots & \mathbf{a}_{j_m} \end{pmatrix} \quad \text{and} \quad A_N = \begin{pmatrix} \mathbf{a}_{k_1} & \mathbf{a}_{k_2} & \cdots & \mathbf{a}_{k_{n-m}} \end{pmatrix}. \tag{1.12}$$

We note that A_B is an invertible $m \times m$ matrix and A_N is an $m \times (n-m)$ matrix. Further, we see that $\mathbf{x} \in \Sigma(\mathbf{b})$ if and only if

$$A_B \mathbf{x}_B + A_N \mathbf{x}_N = \mathbf{b}, \tag{1.13}$$

whence

$$\mathbf{x}_B = A_B^{-1} \left(\mathbf{b} - A_N \mathbf{x}_N \right) = \mathbf{v}_B - A_B^{-1} A_N \mathbf{x}_N, \tag{1.14}$$

since

$$\mathbf{v}_B = A_B^{-1} \mathbf{b}. \tag{1.15}$$

Further, setting

$$\mathbf{c}_B = \begin{pmatrix} c_{j_1} \\ c_{j_2} \\ \vdots \\ c_{j_m} \end{pmatrix} \in \mathbb{R}^m \quad \text{and} \quad \mathbf{c}_N = \begin{pmatrix} c_{k_1} \\ c_{k_2} \\ \vdots \\ c_{k_{n-m}} \end{pmatrix} \in \mathbb{R}^{n-m}, \tag{1.16}$$

we obtain

$$f(\mathbf{x}) = \mathbf{c}_B^T \mathbf{x}_B + \mathbf{c}_N^T \mathbf{x}_N \tag{1.17}$$

and we note that

$$f(\mathbf{v}) = \mathbf{c}_B^T A_B^{-1} \mathbf{b}. \tag{1.18}$$

Thus, substituting (1.14) in (1.17), we find

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{c}_B^T \left(\mathbf{v}_B - A_B^{-1} A_N \mathbf{x}_N \right) + \mathbf{c}_N^T \mathbf{x}_N & (1.19) \\ &= f(\mathbf{v}) + \left(\mathbf{c}_N^T - \mathbf{c}_B^T A_B^{-1} A_N \right) \mathbf{x}_N & (1.20) \\ &\equiv f(\mathbf{v}) + \mathbf{r}^T \mathbf{x}_N, & (1.21) \end{aligned}$$

where

$$\mathbf{r} = \mathbf{c}_N - A_N^T A_B^{-T} \mathbf{c}_B. \tag{1.22}$$

[Here A_B^{-T} is shorthand for the transpose of A_B^{-1}.]

Equation (1.22) provides a simple test for optimality.

Proposition 1.5. If $\mathbf{r} \geq 0$, then $\mathbf{v}$ is an optimal vertex.

Proof. If $\mathbf{x}_N \geq 0$, then (1.22) implies $\mathbf{r}^T \mathbf{x}_N \geq 0$, so that $f(\mathbf{x}) \geq f(\mathbf{v})$. □

Thus, if $\mathbf{r}$ is *not* a vector with non-negative components, then we are at a suboptimal vertex. In order to move to a new vertex, the simplex method chooses

$$\mathbf{x}_N = \theta \mathbf{e}_i, \qquad \theta \geq 0, \tag{1.23}$$

where $i \in \{1, 2, \ldots, n-m\}$ is an integer for which $r_i < 0$. In this case, we obtain

$$\mathbf{x}_B = \mathbf{v}_B - \theta A_B^{-1} A_N \mathbf{e}_i \tag{1.24}$$

and, because the ith component of $\mathbf{r}$ is negative, (1.21) implies

$$f(\mathbf{x}) = f(\mathbf{v}) + \theta \mathbf{r}^T \mathbf{e}_i = f(\mathbf{v}) + \theta r_i < f(\mathbf{v}), \tag{1.25}$$

for $\theta > 0$. Of course, we must ensure $\mathbf{x} \in S$. In other words, we must pick $\theta > 0$ so that

$$\mathbf{x}_B = \mathbf{v}_B - \theta A_B^{-1} A_N \mathbf{e}_i \geq 0. \tag{1.26}$$

Thus, setting

$$\mathbf{d} = A_B^{-1} A_N \mathbf{e}_i, \tag{1.27}$$

we require $\mathbf{v}_B - \theta \mathbf{d} \geq 0$, i.e.

$$(\mathbf{v}_B)_j - \theta d_j \geq 0, \qquad \text{for } 1 \leq j \leq n-m. \tag{1.28}$$

Of course, the larger the permitted value of θ, the greater the reduction in $f(\mathbf{x})$. Thus we set

$$\theta^* = \sup\{\theta > 0 : \mathbf{v}_B - \theta \mathbf{d} \geq 0\} \tag{1.29}$$

or

$$\theta^* = \min\{(\mathbf{v}_B)_k/d_k : d_k > 0\}. \tag{1.30}$$

Then the resulting vector $\mathbf{w}$, where

$$\mathbf{w}_B = \mathbf{v}_B - \theta^*\mathbf{d} \text{ and } \mathbf{w}_N = \theta^*\mathbf{e}_i, \tag{1.31}$$

is a vertex for which $f(\mathbf{w}) < f(\mathbf{v})$. Further, in an attempt to optimize our reduction in f, the simplex method chooses i so that the ith component of $\mathbf{r}$ is maximally negative, i.e. $r_i \le r_k$, for $1 \le k \le n-m$. This motion between vertices is called the simplex method.

Fix for degenerate vertices
The above description is valid when $\mathbf{v}$ is a nondegenerate vertex of S, that is, when $B(\mathbf{v})$ contains m numbers. However, if $\mathbf{v}$ is a degenerate vertex, then there is a fix: We now let B^* be any set of m numbers containing $B(\mathbf{v})$ such that the corresponding columns $\{\mathbf{a}_k : k \in B^*\}$ of A are linearly independent. We then continue the algorithm as given above, **BUT** you are warned that degenerate vertices can cause problems.

Algorithm 1.1. The Simplex Algorithm.

(i) Pick a vertex $\mathbf{v} \in S$. [See the next section for finding vertices.]

(ii) Let $B \equiv B(\mathbf{v})$ be the basic set, $N = \{1, 2, \ldots, n\} \setminus B$ the nonbasic set, proceeding as described at the end of the previous section if $\mathbf{v}$ is a degenerate vertex. Construct $A_B, A_N, \mathbf{c}_B, \mathbf{c}_N, \mathbf{v}_B$ as above.
Let
$$\mathbf{r} = \mathbf{c}_N - A_N^T A_B^{-T}\mathbf{c}_B.$$
If $\mathbf{r} \ge 0$, then STOP: the algorithm has found $\mathbf{v}$ an optimal vertex.

(iii) Let r_i be any maximally negative component of $\mathbf{r}$, that is
$$r_i \le r_j, \qquad 1 \le j \le n-m,$$
and set
$$\mathbf{d} = A_B^{-1}A_N\mathbf{e}_i.$$
Let
$$\theta^* = \min\{(\mathbf{v}_B)_k/d_k : d_k > 0\}.$$
Then the next vertex is given by where
$$\mathbf{x}_B = \mathbf{v}_B - \theta^*\mathbf{d} \quad \text{and} \quad \mathbf{x}_N = \theta^*\mathbf{e}_i.$$
Now let $\mathbf{v}$ denote this new vertex and set $B \equiv B(\mathbf{v})$ and $N \equiv \{1, \ldots, n\} \setminus B(\mathbf{v})$. **GOTO** (ii).

1.3. Some Convex Geometry

This section introduces some topics from the important field of *convex analysis* that yield some useful geometric insights.

Definition 1.6. For any points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p \in \mathbb{R}^n$ and numbers $t_1, t_2, \ldots, t_p \in [0,1]$ satisfying

$$\sum_{k=1}^{p} t_k = 1,$$

the linear combination

$$\sum_{k=1}^{p} t_k \mathbf{x}_k$$

is called a *convex combination* of $\mathbf{x}_1, \ldots, \mathbf{x}_p$.

Example 1.2. The set of all convex combinations of $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ is precisely the line segment joining $\mathbf{x}$ and $\mathbf{y}$; it's the set

$$\{t\mathbf{x} + (1-t)\mathbf{y} : 0 \le t \le 1\}.$$

Definition 1.7. A subset K of $\mathbb{R}^n$ is called *convex* if $\mathbf{x}, \mathbf{y} \in K$ implies every convex combination of $\mathbf{x}$ and $\mathbf{y}$ is also an element of K. Thus the line segment joining any pair of points in K is also contained in K.

Lemma 1.6. The feasible set defined by equation (1.5) is convex.

Proof. Let $\mathbf{z} = t\mathbf{x} + (1-t)\mathbf{y}$. If $\mathbf{x}, \mathbf{y} \in \Sigma(\mathbf{b})$, then $A\mathbf{z} = tA\mathbf{x} + (1-t)A\mathbf{y} = b$ for *every* $t \in \mathbb{R}$. Further, if $\mathbf{x} \ge 0$ and $\mathbf{y} \ge 0$, then, for $t \in [0,1]$, we have

$$z_j = tx_j + (1-t)y_j \ge \min\{x_j, y_j\} \ge 0, \qquad 1 \le j \le n.$$

Thus S is convex. $\square$

Exercise 1.3. Let $K_1, K_2, \ldots, K_N$ be convex subsets of $\mathbb{R}^n$. Then their intersection

$$K = \bigcap_{j=1}^{N} K_j$$

is also a convex set.

Definition 1.8. Let K be a convex subset of $\mathbb{R}^n$. A point $\mathbf{v} \in K$ is called an *extreme point* of K if there do not exist distinct points $\mathbf{x}, \mathbf{y} \in K$ and $t \in (0,1)$ such that $\mathbf{z} = t\mathbf{x} + (1-t)\mathbf{y}$. In other words, an extreme point cannot lie in the interior of a line segment contained in K.

Example 1.3. Consider the tetrahedron

$$K = \{(x,y,z) \in \mathbb{R}^3 : x + y + z \le 1 \text{ and } x \ge 0, y \ge 0, z \ge 0\}.$$

Then the extreme points of K are the corners $(1,0,0)$, $(0,1,0)$ and $(0,0,1)$.

We see that the extreme points in the last example are exactly the vertices defined in the previous section. This is true in general.

Theorem 1.7. *The extreme points of the feasible set S are exactly the vertices of S.*

Proof. Suppose $\mathbf{z} \in S$ is not an extreme point. Then there are distinct points $\mathbf{x}, \mathbf{y} \in S$, and a number $t \in (0,1)$, such that $\mathbf{z} = t\mathbf{x} + (1-t)\mathbf{y}$. Hence $B(\mathbf{x}) \subset B(\mathbf{z})$ and $B(\mathbf{y}) \subset B(\mathbf{z})$. Thus

$$\sum_{k \in B(\mathbf{z})} (x_k - y_k)\,\mathbf{a}_k = 0$$

and the vectors $\{\mathbf{a}_k : k \in B(\mathbf{z})\}$ are linearly dependent, which implies that $\mathbf{z}$ is not a vertex.

Conversely, suppose $\mathbf{x} \in S$ is not a vertex. Thus we may apply the construction of Lemma 1.1 to obtain a nonzero vector $\mathbf{y}$ such that $\mathbf{x} - t\mathbf{y}$ is a feasible point for all sufficiently small $|t|$. Thus there is a number $\theta > 0$ such that $\mathbf{x} + \theta\mathbf{y}$, $\mathbf{x} - \theta\mathbf{y}$ are feasible points, so that

$$\mathbf{x} = \frac{1}{2}\,(\mathbf{x} + \theta\mathbf{y}) + \frac{1}{2}\,(\mathbf{x} - \theta\mathbf{y})$$

and $\mathbf{x}$ is not an extreme point, being the midpoint of the line segment joining the feasible points $\mathbf{x} \pm \theta\mathbf{y}$. □

Definition 1.9. Let A be any subset of $\mathbb{R}^n$. Then the *convex hull* $K(A)$ of A is the set generated by all convex combinations of points of A.

It can be shown that $K(A)$ is the *smallest* convex set containing A. In other words, if C is a convex set such that $A \subset C$, then $K \subset C$. Further, there is a simple link between convex sets and their extreme points.

Proposition 1.8. *Let C be any convex subset of $\mathbb{R}^n$. Let E denote the set of all extreme points of C. Then $C = K(E)$.*

Proof. This proof is not examinable. □

Applying this proposition to the feasible set S of a linear programming problem, we get another proof of the fact that, if there's an optimal feasible point, then there's an optimal vertex. For, if $\mathbf{x} \in S$ is an optimal feasible point, then we can express $\mathbf{x}$ as a convex combination of vertices of S:

$$\mathbf{x} = \sum_{k=1}^{p} w_k \mathbf{v}_k.$$

Thus

$$f(\mathbf{x}) = \sum_{k=1}^{p} w_k f(\mathbf{v}_k) \geq \min\{f(\mathbf{v}_k) : 1 \leq k \leq p\} \geq \min\{f(\mathbf{v}) : \mathbf{v} \text{ is a vertex of } S\}.$$

This leads us to another interesting question: how large can the number p be in the previous equation? It is plausible that, in $\mathbb{R}^2$, any point in the convex hull of a finite set of points is in the triangle generated as the convex hull of some three of the points. This is indeed the case.

Theorem 1.9. (Carathéodory's Theorem) Let A be any subset of $\mathbb{R}^n$. Then every point of the convex hull $K(A)$ is a convex combination of at most $n+1$ points of A.

Proof. Suppose

$$\mathbf{z} = \sum_{k=0}^{p} w_k \mathbf{a}_k,$$

where $p > n$, $w_0, w_1, \ldots, w_p$ are positive numbers, $\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_p \in A$ and $\sum_{k=0}^{p} w_k = 1$. Then

$$0 = \sum_{k=0}^{p} w_k \boldsymbol{\alpha}_k,$$

where $\boldsymbol{\alpha}_k = \mathbf{a}_k - \mathbf{z}$. Thus the vectors $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_p$ are linearly dependent (because any $p > n$ vectors in $\mathbb{R}^n$ are linearly dependent). Hence there are real numbers $y_1, \ldots, y_p$, not all of which vanish, for which

$$\sum_{k=1}^{p} y_k \boldsymbol{\alpha}_k = 0.$$

If we now define $y_0 \equiv 0$, then we see that

$$\sum_{k=0}^{p} (w_k - t y_k)\, \boldsymbol{\alpha}_k = 0$$

for every $t \in \mathbb{R}$. Now consider the kth coefficient

$$c_k(t) = w_k - t y_k$$

as a function of t. We note that $c_0(t) \equiv w_0 > 0$, for all t, and $c_k(0) = w_k > 0$. Thus $c_k(t) > 0$ for all sufficiently small $|t|$, because we have finitely many continuous functions. Of course, if $y_k \neq 0$, then $t_k = w_k/y_k$ is the unique zero of the linear function $c_k(t)$. Let t^* be the smallest modulus element of the zeros of $c_0(t), \ldots, c_p(t)$. Then $c_k(t^*) \geq 0$, for $k = 0, 1, \ldots, p$, and at least one of the numbers $\{c_k(t^*) : 0 \leq k \leq p\}$ must be equal to zero (if not, we could increase t^* in modulus without causing one of the coefficient functions to become negative). Therefore

$$0 = \sum_{k=0}^{p} c_k(t^*)\boldsymbol{\alpha}_k = \left(\sum_{k=0}^{p} c_k(t^*)\mathbf{a}_k\right) - \left(\sum_{k=0}^{p} c_k(t^*)\right)\mathbf{z},$$

or

$$\mathbf{z} = \sum_{k=0}^{p} w_k^* \mathbf{a}_k,$$

where

$$w_k^* = \frac{c_k(t^*)}{\sum_{\ell=0}^{p} c_\ell(t^*)}, \qquad k = 0, 1, \ldots, p,$$

and at least one w_k^* must vanish. Thus we have expressed $\mathbf{z}$ as a convex combination of at most p points in A.

We repeat the construction until $p \le n$, so proving the theorem. □

1.4. Variations on the standard form

At first sight, the LP problem defined by (1.1) seems far more general than the LP problem given by (1.2). However, this seemingly greater generality is illusory. To give some indication of this, we show how several LP problems can be reduced to standard form.

First note that, for the feasible region defined by (1.1) to be nonempty, the number M of linear equality constraints cannot exceed the total number of variables n. If $M = n$, then either S is empty, or it contains exactly one point, namely the solution to the n linear equations. Thus it is natural to exclude this rather trivial case by stipulating $M < n$. Further, if some of the M linear equations can be written as a linear combination of the others, then we can reduce the total number M of linear equations. Therefore we assume that any such reduction has already been effected, leaving M linearly independent equations.

We can now reduce (1.1) to standard form by introducing P *slack variables*

$$x_{n+1}, x_{n+2}, \ldots, x_{n+P}.$$

Specifically, we solve the augmented LP problem

$$\begin{aligned} \text{Minimize} \quad & \mathbf{c}_{\text{aug}}^T \mathbf{x}_{\text{aug}}, \qquad \mathbf{x}_{\text{aug}} \in \mathbb{R}^{n+P} \\ \text{subject to} \quad & m_j(\mathbf{x}_{\text{aug}}) = 0, \qquad 1 \le j \le M, \\ \text{and} \quad & m_{M+j}(\mathbf{x}_{\text{aug}}) = x_{n+j}, \qquad 1 \le j \le P. \end{aligned} \tag{1.32}$$

Here

$$\mathbf{x}_{\text{aug}} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n+P} \end{pmatrix} \text{ and } m_j(\mathbf{x}_{\text{aug}}) = \begin{pmatrix} x_1 & \cdots & x_n \end{pmatrix}^T \mathbf{a}_j - b_j, \qquad 1 \le j \le M+p.$$

1.5. Finding an initial vertex

In some problems, finding an initial vertex is trivial. For example, if every linear equality constraint results from creating a slack variable, then we can simply begin with the slack variables being the basic variables. However, in general this is not the case. However, it is possible to find a vertex of S by applying the simplex algorithm to solve a closely related LP problem; this is the "first phase", the "second phase" being the simplex algorithm starting with the vertex of S located by the first phase.

Algorithm 1.2. Phase I: This algorithm finds a vertex of S provided that S is nonempty. Phase II: We then use this vertex to solve the original LP problem.

(i) Let $\sigma_k = \text{sign}(b_k)$, for $1 \le k \le m$, and define the diagonal matrix

$$D = \begin{pmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_m \end{pmatrix}.$$

Replace A by DA and $\mathbf{b}$ by $D\mathbf{b}$. Thus we have simply multiplied every linear equality constraint by ± 1 so as to obtain $\mathbf{b} \ge 0$.

(ii) Use the simplex method to solve the LP problem

$$\begin{aligned} \text{Minimize} \quad & r_1 + r_2 + \cdots + r_m \\ \text{subject to} \quad & A\mathbf{x} + \mathbf{r} = \mathbf{b} \\ \text{and} \quad & \mathbf{x} \ge 0, \mathbf{r} \ge 0, \end{aligned} \tag{1.33}$$

starting with the vertex $\mathbf{x} = 0$, $\mathbf{r} = \mathbf{b}$. The simplex algorithm will terminate with $\mathbf{r} = 0$ and $\mathbf{x}$ a vertex of S,

(iii) Use the newly computed vertex as the initial vertex for the simplex method applied to the original problem.

Note that the LP problem defined by (1.33) has $m + n$ variables and m linear constraints. Thus finding a vertex is just as difficult as solving the LP problem once an initial vertex is known.

1.6. Some computational points

Older treatments of the simplex method were designed for readers who may not have met matrices during their undergraduate career. Therefore such expositions introduce matrices tacitly, avoiding almost all explicit mention of matrix algebra. The matrix A is written out in full and is called the "tableau". This is the reason for the ubiquity of tableaux in many texts, although this exposition is far beyond its "use by" date. Today, every student in every numerate discipline learns about matrices from the beginning of their undergraduate career. Therefore we say nothing about tableaux in this treatment. If you want to see how it used to be done, then look up the relevant sections of Strang.

One disadvantage of the simplex method as described here is the seeming need to compute A_B^{-1} at every step of the simplex method. This can be avoided once we observe that the basic set changes by one element only from one iteration to the next. Changing the order of the elements in the basic set as necessary, we can assume that only the qth column changes, that is

$$A_B^{\mathrm{new}} = A_B^{\mathrm{old}} + (\mathbf{b} - \mathbf{a})\,\mathbf{e}_q^T, \tag{1.34}$$

where $\mathbf{a}$ is the qth column of A_B^{old} and $\mathbf{b}$ is the qth column of A_B^{new}. This relies on the fact that, for any two (column) vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$, the matrix product $\mathbf{u}\mathbf{v}^T$ is an $n \times n$ matrix whose (j,k)th element is $u_j v_k$. Equation (1.34) is called a *rank one update*, because any matrix of the form $\mathbf{u}\mathbf{v}^T$ has rank one. Indeed, we have

$$\left(\mathbf{u}\mathbf{v}^T\right)\mathbf{x} = \mathbf{u}\left(\mathbf{v}^T\mathbf{x}\right),$$

by associativity of matrix multiplication. Thus $\mathbf{u}\mathbf{v}^T$ maps every vector in $\mathbb{R}^n$ to a multiple of $\mathbf{u}$ and the matrix $\mathbf{u}\mathbf{v}^T$ has rank one.

The ingenious *Sherman-Morrison* formula allows us to avoid the recomputation of the inverse when a rank one change occurs.

Theorem 1.10. Let $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ and let E be any invertible $n \times n$ matrix whose inverse has already been computed. If

$$F = E + \mathbf{u}\mathbf{v}^T \tag{1.35}$$

and $\mathbf{v}^T E^{-1}\mathbf{u} \neq -1$, then F is invertible and

$$F^{-1} = E^{-1} - \left(\frac{\boldsymbol{\alpha}\boldsymbol{\beta}^T}{1 + \mathbf{v}^T\boldsymbol{\alpha}}\right), \tag{1.36}$$

where

$$\boldsymbol{\alpha} = E^{-1}\mathbf{u} \qquad \text{and} \qquad \boldsymbol{\beta} = E^{-T}\mathbf{v}. \tag{1.37}$$

These calculations require a multiple of n^2 arithmetic operations.

Proof. We compute the product of F and the claimed formula for its inverse.

$$\begin{aligned}(E+\mathbf{u}\mathbf{v}^T)\left(E^{-1}-\frac{E^{-1}\mathbf{u}\mathbf{v}^TE^{-1}}{1+\mathbf{v}^TE^{-1}\mathbf{u}}\right) &= I+\mathbf{u}\mathbf{v}^TE^{-1}-\frac{\mathbf{u}\mathbf{v}^TE^{-1}}{1+\mathbf{v}^TE^{-1}\mathbf{u}}-\frac{\mathbf{u}\mathbf{v}^TE^{-1}\mathbf{u}\mathbf{v}^TE^{-1}}{1+\mathbf{v}^TE^{-1}\mathbf{u}}\\ &= I+\mathbf{u}\mathbf{v}^TE^{-1}-\left(\frac{\mathbf{u}\mathbf{v}^TE^{-1}+(\mathbf{v}^TE^{-1}\mathbf{u})\mathbf{u}\mathbf{v}^TE^{-1}}{1+\mathbf{v}^TE^{-1}\mathbf{u}}\right)\\ &= I+\mathbf{u}\mathbf{v}^TE^{-1}-\mathbf{u}\mathbf{v}^TE^{-1}\end{aligned} \tag{1.38}$$

The main computation required is to multiply an $n\times n$ matrix by two vectors in $\mathbb{R}^n$, which requires a multiple of n^2 operations. □

You have seen a more general version of this formula in the assessed work. The formula is useful because computing the inverse of A_B^{new} afresh on each iteration would require a multiple of n^3 operations — consider the saving when, say, $n = 100$ or $n = 10000$.

Karmarkar's Algorithm: The simplex method achieved prominence in the 1950s when tests comparing its performance to several other methods for solving LP problems conclusively demonstrated its great superiority. Its creator, George Dantzig, is still alive and famous, and the ambitious student might care to read his classic textbook. The simplex method and its many variants then dominated the theory and practice of linear programming for some thirty years

In the 1960s and 1970s, several mathematicians began asking, and answering, some interesting complexity questions. Specifically, their interest concerned the questions

(i) What's the worst possible performance of the simplex algorithm.
(ii) How does it behave on average?

Of course, enormous experience in the use of the simplex algorithm had by then been developed, so that it was conjectured that the number of vertices visited was at most a linear function of the number of variables n and the number m of linear equality constraints, because this was the observed behaviour of the algorithm. However, this is quite wrong, as was found by Klee, Minty and Chvatal in the early 1970s. Their example is simple to describe:

$$\begin{aligned}&\text{Maximize} && \sum_{j=1}^{n}10^{j-1}x_j\\ &\text{subject to} && x_i+2\sum_{j=i+1}^{n}10^{j-i}x_j\le 10^{2n-2i}, \qquad i=1,2,\ldots,n\\ &\text{and} && \mathbf{x}\ge 0.\end{aligned} \tag{1.39}$$

Exercise 1.4. Write down the Klee-Minty-Chvatal problem when $n = 5$.

Thus the Klee-Minty example is an n-dimensional problem with n linear inequality constraints. The feasible region is a cuboid with 2^n extreme points. It can be shown that, if the initial vertex is $\mathbf{x} = 0$, then the simplex algorithm visits *all* 2^n

vertices before reaching the minimal vertex. Thus the simplex algorithm requires time growing exponentially with n. Apparently this behaviour has never been discovered in practice, so that such simplexes are presumably rare. Therefore, by making some probabilistic assumptions, many mathematicians began to attempt to show that the average case behaviour is as observed, with limited success. In 1979, Khachiyan, a Russian mathematician, constructed a new method which he proved would reach the minimal vertex in time given by a polynomial function of m and n, but this still far exceeds the observed linearity of the simplex method. Further, their new algorithm was greatly inferior to the simplex method in performance. Then, in 1984, Narendra Karmarkar, a mathematician employed by AT&T, presented a new algorithm for which he claimed performance superior to the simplex method. The practical importance of linear programming problems caused his discovery to reach the front page of the *New York Times*, but his work generated much controversy. Specifically, although his algorithm was clear and geometrically highly attractive (it goes through the middle of the simplex rather than staying on the edges), its claimed performance was dependent on numerical methods being used in intermediate calculations whose details were known only to AT&T. Since AT&T refused to make these details public knowledge, Karmarkar's claims could not be verified. However, once the arguments had died down in the late 1980s, the consensus was that his method was superior to the simplex algorithm for certain classes of LP problems, but by no means for all. Further, in 1985, it was found that Karmarkar's method was a special case of a more general class of algorithms popular in the 1960s for finding local minima of a general nonlinear function subject to nonlinear constraints. For various reasons, these algorithms were generally abandoned in the 1970s, but their identification with Karmarkar's method rejuvenated the research field. This is now a highly active area of research, of great interest to pure mathematicians, applied mathematicians, computer scientists and economists.

Some of the details of Karmarkar's method can be found in Strang's books. Any student should be able to follow Strang's relaxed exposition without too much trouble.

1.7. Convexity and the Gauss-Lucas Theorem

In this section we derive a beautiful convexity result.

Theorem 1.11. (Gauss-Lucas Theorem) Let $p(z)$ be a polynomial of degree n with distinct roots

$$Z = \{z_1, z_2, \ldots, z_m\} \subset \mathbb{C}.$$

Then every root of its derivative $p'(z)$ is an element of the convex hull $K(Z)$.

Before proving Theorem 1.11, we state some corollaries and preliminary results.

Corollary 1.12. If every root of a polynomial has non-negative real part, then every root of its derivative also has non-negative real part.

Corollary 1.13. If every root of a polynomial lies in a convex set K, then every root of its derivative is an element of K.

Proof. Both corollaries rely on the fact that $Z \subset K$ implies $K(Z) \subset K$. □

These corollaries are surprisingly useful in their own right.

To prove Theorem 1.11, we need some notation. For any complex sequence $\alpha_1, \alpha_2, \ldots, \alpha_N$, we define

$$\prod_{k=1}^{N} \alpha_k \equiv \alpha_1 \alpha_2 \cdots \alpha_N.$$

Thus this notation plays the corresponding role of Σ when studying products.

Exercise 1.5. If $\alpha_1, \ldots, \alpha_N$ are positive numbers, then

$$\log_e \left(\prod_{j=1}^{N} \alpha_j \right) = \sum_{j=1}^{N} \log_e \alpha_j.$$

The Fundamental Theorem of Algebra (hopefully proved in your Complex Analysis course) asserts that every polynomial $p(z)$ of degree n can be written as

$$p(z) = \lambda \prod_{k=1}^{m} (z - z_k)^{\nu_k}, \qquad z \in \mathbb{C}. \tag{1.40}$$

where $\lambda \in \mathbb{C}$ is a constant and $\nu_1, \ldots, \nu_m$ are positive integers satisfying $\nu_1 + \cdots + \nu_m = n$. There is a simple formula for the derivative $p'(z)$:

Lemma 1.14.

$$p'(z) = \lambda \sum_{j=1}^{m} \nu_j (z - z_j)^{\nu_j - 1} \prod_{k=1, k \neq j}^{m} (z - z_k)^{\nu_k}. \tag{1.41}$$

Proof. Direct differentiation of the product. □

It will come as no surprise to any student of the residue theorem that it is easier to study

$$q(z) \equiv \frac{p'(z)}{p(z)}. \tag{1.42}$$

Further, we see that

$$q(z) = \sum_{j=1}^{m} \frac{\nu_j}{z - z_j}. \tag{1.43}$$

Proof. . . . *of Theorem 1.11*: Let w be any zero of p'. *Either* (i) $p(w) = p'(w) = 0$, *or* (ii) $p(w) \neq 0$ and $p'(w) = 0$. In case (i), we have $w \in Z \subset K(Z)$, as required. In the latter case, we obtain

$$0 = q(w) = \sum_{j=1}^{m} \frac{\nu_j}{w - z_j} = \sum_{j=1}^{m} \frac{\nu_j \left(\overline{w} - \overline{z_j}\right)}{|w - z_j|^2}. \tag{1.44}$$

Hence, taking complex conjugates of both sides,

$$w \sum_{k=1}^{m} \nu_k |w - z_k|^{-2} = \sum_{j=1}^{m} z_j \nu_j |w - z_j|^{-2}, \tag{1.45}$$

or

$$w = \sum_{j=1}^{m} \tau_j z_j, \tag{1.46}$$

where

$$\tau_j = \frac{\nu_j |w - z_j|^{-2}}{\sum_{k=1}^{m} \nu_k |w - z_k|^{-2}}. \tag{1.47}$$

Thus $\tau_j > 0$ and $\tau_1 + \cdots + \tau_m = 1$, and w is indeed a convex combination of the roots of $p(z)$. $\square$

The following exercise displays a typical use of the Gauss-Lucas Theorem.

Exercise 1.6. Prove that the diameter of the set of roots of the quartic polynomial $z^4 + az^3 + bz^2 + cz + d$ is at least $\sqrt{(a^2/4) - (2b/3)}$.

2. Networks

My treatment is based on that of T. W. Körner, *The Pleasures of Counting*, which I highly recommend as a fascinating general text.

2.1. Introduction

We imagine n computer centres, or *nodes* $\{1, 2, \dots, n\}$ linked by routes, or *edges.* The edge linking nodes i and j has capacity c_{ij} bits per second, where $c_{ij} \geq 0$. We don't assume that $c_{ji} = c_{ij}$. Our problem is to send as many bits per second as possible from node 1, the *source*, to node n, the *sink.* Thus we must specify the number of bits per second x_{ij} sent from node i to node j, where

$$0 \leq x_{ij} \leq c_{ij}, \qquad \text{for } 1 \leq i, j \leq n. \tag{2.1}$$

Further, we shall assume $c_{jj} = 0$ and that messages can only be created or destroyed at nodes 1 and n. Thus we have the *conservation conditions*

$$\sum_{k=1}^{n} x_{jk} - x_{kj} = 0, \qquad \text{for } 1 < j < n. \tag{2.2}$$

We see that

$$\sum_{k=1}^{n} x_{jk}$$

is the net number of bits per second leaving node j whilst

$$\sum_{k=1}^{n} x_{kj}$$

is the net numbers of bits per second reaching node j.

The conservation conditions imply that the total number of messages reaching node n each second is given by

$$F = \sum_{k=1}^{n} x_{1k} - x_{k1}, \tag{2.3}$$

and we shall call this the *flow value.* Another consequence of the (2.2) is the relation

$$F = \sum_{k=1}^{n} x_{kn} - x_{nk}. \tag{2.4}$$

Thus our problem is to maximize the flow value F given by (2.3) subject to the constraints (2.1,2.2). We see that our task is to minimize a linear function subject to some linear equality and inequality constraints. Specifically, our network problem is as follows.

$$\begin{aligned} \text{Minimize} \quad & F(\mathbf{x}) = \sum_{k=1}^{n} (x_{1k} - x_{k1}) \\ \text{subject to} \quad & \sum_{k=1}^{n} (x_{jk} - x_{kj}) = 0, \qquad \text{for } 1 < j < n \\ \text{and} \quad & 0 \le x_{jk} \le c_{jk}, \qquad 1 \le j,k \le n. \end{aligned} \tag{2.5}$$

Here

$$\mathbf{x} = \begin{pmatrix} x_{11} \\ x_{12} \\ \vdots \\ x_{1n} \\ x_{21} \\ \vdots \\ x_{n-1\ n} \end{pmatrix} \in \mathbb{R}^{n^2}.$$

Definition 2.1. (i) Any feasible vector $\mathbf{x} \in \mathbb{R}^{n^2}$ will be called a *flow.*
(ii) The *zero flow* is simply the choice $\mathbf{x} = 0$.
(iii) A flow $\mathbf{x}^*$ is *maximal*, or *optimal*, if $F(\mathbf{x}) \le F(\mathbf{x}^*)$ for every flow $\mathbf{x}$. There may be more than one maximal flow.

We could solve (2.5) using the simplex method. Unfortunately the large number of slack variables required renders the basic form of the simplex method rather inefficient. Our main aim in this section is to study a variant of the simplex method called the *Ford-Fulkerson* algorithm.

2.2. The Ford-Fulkerson Algorithm

Lemma 2.1. Let $\mathbf{x}$ be a suboptimal flow for (2.4) and let us define subsets $A_0, A_1, \ldots$ of $\{1, 2, \ldots, n\}$ recursively, as follows. Let

$$A_0 = \{1\} \tag{2.6}$$

$$A_{k+1} = A_k \cup \{j \in [1,n] : c_{ij} > x_{ij} - x_{ji} \text{ for some } i \in A_k\}, \qquad k \ge 0. \tag{2.7}$$

Then

(i) **either** there exists $r \ge 1$ such that $n \in A_r$
(ii) **or** there exists $r \ge 1$ such that $n \notin A_r$ and $A_r = A_{r+1} = A_{r+2} = \cdots$.

Proof. If $A_k \ne A_{k+1}$, then $|A_{k+1}| \ge |A_k| + 1$. Since there are only n nodes, the sets $A_0 \subset A_1 \subset \cdots$ either reach node n or cease to grow when (say) $k = r$. □

Lemma 2.1 is the crux of the Ford-Fulkerson Algorithm. We shall some more notation.

Definition 2.2. Let $U \subset \{1, 2, \ldots, n\}$ satisfy $1 \in U$ and $n \notin U$, and let $V = \{1, 2, \ldots, n\} \setminus U$ (so $n \in V$). Then we say that $C = (U, V)$ defines a *cut* in the network. The *cut capacity* $\operatorname{cap}(C)$ (or *cut value*) is simply the sum of the capacities of all edges joining nodes in U to nodes in V, that is

$$\operatorname{cap}(C) = \sum_{i \in U, j \in V} c_{ij}. \tag{2.8}$$

Proposition 2.2. For any cut $C = (U, V)$ and any flow $\mathbf{x}$, we have

$$F(\mathbf{x}) \le \operatorname{cap}(C). \tag{2.9}$$

Proof. This is the elementary inequality

$$\operatorname{cap}(C) = \sum_{i \in U, j \in V} c_{ij} \ge \sum_{i \in U, j \in V} x_{ij} \ge \sum_{i \in U, j \in V} x_{ij} - x_{ji} = F(\mathbf{x}).$$

□

Corollary 2.3. (i) The minimum cut capacity is at least the maximum flow value.
(ii) If C is a cut and $\mathbf{x}$ is a flow satisfying $\operatorname{cap}(C) = F(\mathbf{x})$, then C is a minimal cut and $\mathbf{x}$ is a maximal flow.

Proof. These are immediate consequences of Proposition 2.2. □

Proposition 2.4. Let $\mathbf{x}$ be a flow and let $A_0 \subset A_1 \subset A_2 \subset \cdots$ be the sets defined in Lemma 2.1. If $n \in A_r$, for some r, then we can construct a new flow $\mathbf{y}$ for which $F(\mathbf{y}) > F(\mathbf{x})$.

Proof. We define a sequence $m_r, m_{r-1}, \ldots, m_1, m_0$ recursively by setting $m_r = n$ and, for $k = r, r-1, \ldots, 1$, we choose m_{k-1} to be any node in A_{k-1} for which

$$c_{m_{k-1}m_k} > x_{m_{k-1}m_k} - x_{m_k m_{k-1}}. \tag{2.10}$$

Thus $m_0 = 1$, because $m_0 \in A_0 = \{1\}$. We have therefore constructed a *flow-augmenting path.* Setting

$$\delta = \min\{c_{m_{k-1}m_k} - \left(x_{m_{k-1}m_k} - x_{m_k m_{k-1}}\right) : k = 1, \ldots, r\} \tag{2.11}$$

we define our new flow $\mathbf{y}$ along the flow-augmenting path by increasing $x_{m_{k-1}m_k}$ by at most δ, or reducing $x_{m_k m_{k-1}}$ by most δ, or some combination of the two. For all other edges the flow $\mathbf{x}$ is unchanged. Thus

$$F(\mathbf{y}) = F(\mathbf{x}) + \delta. \tag{2.12}$$

□

Proposition 2.5. Let $\mathbf{x}$ be a flow and let $A_0 \subset A_1 \subset A_2 \subset \cdots$ be the sets defined in Lemma 2.1. If $n \notin A_r$ and $A_r = A_{r+1}$, then $\mathbf{x}$ is a maximal flow and $C = (A_r, \{1, 2, \ldots, n\} \setminus A_r)$ is a minimal cut.

Proof. If

$$A_{r+1} = A_r \cup \{j \in [1, n] : c_{ij} > x_{ij} - x_{ji}, \text{ for some } i \in A_r\}, \tag{2.13}$$

then $J \notin A_r$ implies $c_{iJ} = x_{iJ} - x_{jI}$, for every $i \in A_r$. Hence $x_{iJ} = c_{iJ}$ and $c_{Ji} = 0$ when $i \in A_r$ and $J \notin A_r$. (In detail, we have

$$c_{iJ} = x_{iJ} - x_{Ji} \le x_{iJ} \le c_{iJ},$$

or $c_{iJ} = x_{iJ}$ and $x_{Ji} = 0$.) Thus, if we choose the cut $C = (A_r\{1, \ldots, n\} \setminus A_r)$, then its capacity satisfies

$$\operatorname{cap}(C) = \sum_{i \in A_r, j \notin A_r} c_{ij} = \sum_{i \in A_r, j \notin A_r} x_{ij} - x_{ji} = F(\mathbf{x}). \tag{2.14}$$

Thus, by Corollary 2.3, $\mathbf{x}$ is a maximal flow and C is a minimal cut. □

The Ford-Fulkerson algorithm is now straightforward to describe.

Algorithm 2.1. The Ford-Fulkerson algorithm.

(i) Choose be any flow $\mathbf{x}$ (the zero flow $\mathbf{x} = 0$ will suffice).
(ii) Construct the sets $A_0 \subset A_1 \subset A_2 \subset \cdots$ defined by (2.7). If there exists r such that $n \notin A_r$ and $A_r = A_{r+1}$, then, by Proposition 2.5, we STOP, because our flow is maximal. Otherwise apply Proposition 2.4 to construct a new flow $\mathbf{y}$ with $F(\mathbf{y}) > F(\mathbf{x})$. Repeat.

This algorithm can fail to converge if the flows and the capacities are real numbers. However, if the flows and capacities are integers, then convergence in finitely many steps follows from the relation

$$F(\mathbf{y}) \ge F(\mathbf{x}) + 1 \tag{2.15}$$

and noting that any cut provides an upper bound on our maximal flow.

Exercise 2.1. Let all flows and capacities be integers and let $M = \cap(C)$ for any cut C. Show that, if the initial flow is the zero flow, then Ford-Fulkerson requires at most M iterations to attain a maximal flow.

3. Basic Graph Theory

There is really no suitable textbook for this part of the course, although some of the material is covered in chapters of the following books, listed roughly in order of difficulty. If you wish to buy one of these, the first is probably most suitable.

(i) G. Strang, *Introduction to Linear Algebra.*
(ii) G. Strang, *Introduction to Applied Mathematics.*
(iii) P. J. Cameron, *Combinatorics.*
(iv) B. Bollobas, *Modern Graph Theory.*

3.1. Fundamentals

Speaking informally, a graph is just some dots, called *nodes* or *vertices*, joined by *edges*. The edges need not be straight lines and are not restricted to lie in two dimensional space. In this section, we shall not view the edges as *directed*, but you have already seen directed graphs in the section on the Ford-Fulkerson algoritm. Further, we require that every edge join different vertices. The formal definition is as follows.

Definition 3.1. A *graph* $G = (V, E)$ is a finite set V, whose elements are called *vertices*, and a subset E of $V \times V$, called *edges*. No edge can be of the form (x, x), $x \in V$. FUrther, if $(x, y) \in E$, then we usually just write xy to denote the edge.

Example 3.1. Let the vertices of our graph be labelled x, y, z and the edges are (x, y), (y, z), (z, x); we have, of course, formed a triangle.

Another way to summarize the information represented by a graph is to form the $n \times n$ matrix A whose elements are either zero or one, and $A_{jk} = 1$ if and only if vertices j and k are joined by an edge. Such a matrix is called an adjacency matrix. We see that the matrix is symmetric (that is, $A_{jk} = A_{kj}$) and its diagonal elements are zero, that is $A_{jj} = 0$, $1 \le j \le n$.

Example 3.2. For the last example, the adjacency matrix is

$$A = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}.$$

Thus all of graph theory reduces to the study of symmetric matrices of zeros and ones whose diagonal elements are zero! It might seem surprising that rich structures of great theoretical and practical interest might arise from such a simple definition, but such is indeed the case.

To discuss properties of graphs we need some terminilogy, much of is straightforward, which we present now.

Definition 3.2. Let $G = G(V, E)$ be a graph.

(i) If the vertices $x, y \in V$ are joined by an edge, i.e. $xy \in E$, then we say that x and y are *adjacent*.

(ii) If we have vertices $x_0, x_1, x_2, \ldots, x_\ell \in V$ such that $x_0x_1, x_1x_2, \ldots, x_{\ell-1}x_\ell \in E$, then we say that we have a *path* of length ℓ; thus the length of a path is simply the number of edges in the path. There is no requirement that the vertices all be different, so that a path can return to a previous vertex.

(iii) If we have vertices $x_1, \ldots, x_\ell \in V$ and edges $x_1x_2, \ldots, x_{\ell-1}x_\ell, x_\ell x_1 \in E$, the we say that we have a *cycle* of length ℓ.

(iv) If every pair of vertices in a graph can be joined by a path, then we say that the graph is *connected*.

(v) A graph without cycles (an *acyclic graph*) is called a *forest*. A connected forest is called a *tree*.

(vi) If G is a connected graph, then a *spanning tree* is an acyclic subgraph that contains every vertex. In other words, it's a maximal acyclic subgraph of G (in the sense that adding one more edge of G creates a cycle).

(vii) We can define a *distance*, or *metric*, on the vertices of the graph by declaring the distance $d(x, y)$ between two vertices $x, y \in V$ to be the length of a shortest path joining the vertices. Further, we set $d(x, x) = 0$ and let $d(x, y) = \infty$ if x and y are not joined by a graph.

(viii) If $x \in G$, then we let

$$\Gamma(x) = \{y \in V : xy \in E\}.$$

In other words, $\Gamma(x)$ is the set of vertices adjacent to x. We define the *degree* $d(x)$ of x to be the number of elements in $\Gamma(x)$, that is

$$d(x) = |\Gamma(x)|.$$

Further, if the vertices of the graph are labelled $\{1, 2, \ldots, n\}$, then we shall write d_k instead of $d(k)$. In terms of the adjacency matrix A for the graph, we have

$$d_j = \sum_{k=1}^{n} A_{jk}, \qquad 1 \le j \le n. \tag{3.1}$$

This notation can also be extended to subsets of the graph. If $S \subset V$, then we let $\Gamma(S)$ denote the set of vertices in G that are adjacent to at least one point of S.

(ix) The total number of edges in the graph will be denote $e(G)$. Thus $e(G) = |E|$.

Proposition 3.1. Let G be a graph. Then

$$\sum_{x \in V} d(x) = 2e(G). \tag{3.2}$$

Proof. If we label the vertices $1, 2, \ldots, n$, then we must show that

$$\sum_{j=1}^{n} d_j = 2e(G).$$

But, because every edge corresponds to two ones in the adjacency matrix (because there's an edge joining vertices j and k if and only if $A_{jk} = A_{kj} = 1$), we have

$$2e(G) = \sum_{j=1}^{n} \sum_{k=1}^{n} A_{jk} = \sum_{j=1}^{n} d_j.$$

□

Proposition 3.2. A graph G can be written as a disjoint union of cycles if and only if every vertex has even degree.

Proof. If the graph is a disjoint union of cycles, then a vertex belonging to k cycles must have degree $2k$.

Conversely, let $x_0 x_1 \ldots x_\ell$ be a path of maximal length in G. Since $d(x_\ell)$ is even, there must be at least one other vertex y adjacent to x_ℓ. But then we must have $y \in \{x_0, x_1, \ldots, x_{\ell-1}\}$, because otherwise the path chosen would not be of maximal length. Thus we have obtained a cycle. Deleting these edges from the graph, we obtain a new graph with strictly fewer edges such that every vertex still has even degree. Thus we can now repeat the construction until all edges have been assigned to cycles. □

The next lemma is the *Cauchy-Schwarz inequality*, a result that's extremely useful in its own right.

Lemma 3.3. Let $x_1, x_2, \ldots, x_n$ and $y_1, y_2, \ldots, y_n$ be real numbers. Then

$$\left(\sum_{j=1}^{n} x_j y_j \right)^2 \le \left(\sum_{k=1}^{n} x_k^2 \right) \left(\sum_{\ell=1}^{n} y_\ell^2 \right).$$

Proof. Define the sum of squares

$$Q(t) = \sum_{k=1}^{n} (y_k - t x_k)^2, \qquad t \in \mathbb{R}.$$

Then $Q(t) \ge 0$, for all $t \in \mathbb{R}$. Expanding, we find the quadratic

$$Q(t) = at^2 + bt + c,$$

where

$$a = \sum_{k=1}^{n} x_k^2, \quad b = -2\sum_{k=1}^{n} x_k y_k \quad \text{and } c = \sum_{k=1}^{n} y_k^2.$$

Thus the discriminant satisfies $b^2 - 4ac \le 0$, which is the required inequality. □

Exercise 3.1. Let $x_1, x_2, \ldots, x_n$ be any n real numbers. Prove that

$$\left(\sum_{k=1}^{n} x_k\right)^2 \le \sum_{k=1}^{n} x_k^2.$$

The following remark yields some geometric insight. If we let

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \quad \text{and} \quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix},$$

then

$$Q(t) = \|\mathbf{y} - t\mathbf{x}\|^2,$$

where $\|\mathbf{u}\|^2 = u_1^2 + u_2^2 + \cdots + u_n^2$.

Exercise 3.2. Prove that equality is attained in the Cauchy-Schwarz inequality if and only if the vectors $\mathbf{x}$ and $\mathbf{y}$ are linearly dependent.

Theorem 3.4. (Mantel's Theorem) Let G be a graph with n vertices. If G is triangle-free, then $e(G) \le n^2/4$.

Proof. If vertices j and k are adjacent, then $\Gamma(j) \cap \Gamma(k)$ is empty, because any vertex adjacent to both j and k would then form a triangle. Thus $d_j + d_k \le n$ if $A_{jk} = 1$. Another way to write this as

$$A_{jk}(d_j + d_k) \le n, \qquad \text{for } 1 \le j, k \le n.$$

Therefore

$$\sum_{j=1}^{n}\sum_{k=1}^{n} A_{jk}(d_j + d_k) \le 2ne(G).$$

In other words,

$$\sum_{j=1}^{n}\sum_{k=1}^{n} A_{jk} \sum_{\ell=1}^{n} A_{j\ell} + A_{k\ell} \le 2ne(G),$$

that is,

$$2\sum_{j=1}^{n} d_j^2 \le 2ne(G).$$

Now the Cauchy-Schwarz inequality tells us that

$$\left(\sum_{j=1}^{n} d_j\right)^2 \le n \sum_{j=1}^{n} d_j^2.$$

Thus

$$(2e(G))^2 \le \sum_{j=1}^{n} d_j^2 \le ne(G),$$

which simplifies to $e(G) \le n^2/4$. □

We now introduce a highly important class of graphs.

Definition 3.3. Let G be a graph for which the two sets V_1, V_2 satisfy $V_1 \cup V_2 = V$, $V_1 \cap V_2 = \emptyset$ and every edge joins a vertex of V_1 to a vertex of V_2. Then we say that the graph is *bipartite*.

Proposition 3.5. A graph is bipartite if and only if it contains no odd cycles.

Proof. Suppose G is bipartite and $x_1 x_2 \cdots x_m$ is a cycle. Without loss of generality, we may assume $x_1 \in V_1$. Thus $x_2 \in V_2$, $x_3 \in V_1$, etc. In other words, we have $x_\ell \in V_2$ if and only if ℓ is even. But x_m is joined to x_1, so that x_m must be a vertex of V_2. Hence m is an even number.

Conversely, suppose G is a graph without odd cycles and pick any vertex $a \in V$. I claim that, if we define

$$V_1 = \{x \in G : d(a, x) \text{ is an odd number } \}$$

and

$$V_2 = \{x \in G : d(a, x) \text{ is an even number } \},$$

then every edge in the graph must join a vertex of V_1 to a vertex of V_2; thus G is bipartite.

Indeed, suppose $x, y \in V_1$ were adjacent. Thus there are paths joining a to x and a to y with, say, lengths p and q respectively, where p and q are odd integers. Therefore, if x and y are adjacent, then we have a cycle whose length is the odd number $p + q + 1$, which is forbidden by hypothesis. The same argument applies if $x, y \in V_2$, except that, in this case, p and q are even numbers. □

Example 3.3. Let G be a graph with vertices $V_1 = \{x_1, \ldots, x_N\}$ and $V_2 = \{y_1, \ldots, y_N\}$ and let each vertex of V_1 be joined to every vertex of V_2. Thus G has $2N$ vertices and N^2 edges. Setting $n = 2N$, we see that $e(G) = n^2/4$ edges, thus attaining the upper bound in Mantel's theorem.

3.2. Hall's Marriage Theorem

Suppose we have n girls $V_1 = \{g_1, \ldots, g_n\}$ and n boys $V_2 = \{b_1, \ldots, b_n\}$. We now create a bipartite graph G from the vertices $V = V_1 \cup V_2$ by joining g_j to b_k by an edge if and only if they are judged compatible. If you wish, you may imagine our task to be the arrangement of marriages in a postapocalyptic world. Our aim is to create as many compatible marriages as possible; we shall say that we have a *complete matching* if it is possible to achieve n marriages between compatible pairs. If a complete matching exists, then every set of r girls must be compatible with as least r boys, for $1 \le r \le n$; more briefly, we shall say that every r girls likes $\ge r$ boys. Remarkably, the converse is also true.

Theorem 3.6. If every r girls likes $\ge r$ boys, for $1 \le r \le n$, then a complete matching is possible.

Proof. We proceed by induction on n, noting that the result is obvious when $n = 1$. Thus let us assume the truth of the theorem for any set of N girls and boys when $N < n$.

Now

(i) **either** every r girls likes at least $r + 1$ boys, for $1 \le r \le n - 1$,

(ii) **or** there is an integer $r < n$ and a set of r girls that like exactly r boys.

In the first case, let's marry any compatible couple, g_1, b_1 say. Then any r girls in the remaining $n - 1$ girls $g_2, \ldots, g_n$ likes at least $r + 1$ boys in $b_1, \ldots, b_n$. In particular, they like at least r boys in $b_2, \ldots, b_n$. We can therefore, by induction hypothesis, obtain a complete matching for $g_2, \ldots, g_n$ and $b_2, \ldots, b_n$, which, together with our marriage of g_1 to b_1, furnishes a complete matching for all n girls and boys.

The second case is slightly more slippery. By relabelling as necessary, we can assume that girls $g_1, g_2, \ldots, g_r$ are compatible only with boys $b_1, \ldots, b_r$. Thus every k girls of $g_1, \ldots, g_r$ must like at least k boys in $b_1, \ldots, b_r$ and, by induction hypothesis, we can construct a complete matching for the first r girls and boys. Now, I claim that any m of the remaining girls $g_{r+1}, \ldots, g_n$ must like at least m of the remaining boys $b_{r+1}, \ldots, b_n$. For, if this were not so, then the $r + m$ girls formed by adjoining girls $g_1, \ldots, g_r$ to the m girls chosen would like strictly fewer than $r + m$ boys, contradicting hypothesis. Thus, by induction hypothesis, we can obtain a complete matching for $g_{r+1}, \ldots, g_n$, $b_{r+1}, \ldots, b_n$ also. □

The proof of Hall's marriage theorem is constructive, in the sense that, given n girls and boys, we could check the required property for every subset of the girls. However, there are $\binom{n}{r}$ ways to pick r girls from n, so that there are, in total,

$$\sum_{r=1}^{n} \binom{n}{r} = 2^n - 1$$

sets to inspect.

Exercise 3.3. Use the binomial expansion

$$(1+t)^n = \sum_{k=0}^{n} \binom{n}{k} t^k$$

to prove that

$$\sum_{k=0}^{n} \binom{n}{k} = 2^n.$$

Further, prove that

$$\binom{n}{0} + \binom{n}{2} + \binom{n}{4} + \cdots = \binom{n}{1} + \binom{n}{3} + \binom{n}{5} + \cdots.$$

A better way to construct a complete matching is to turn the graph of our girls and boys into a network by adding a source node s and a sink node t. We join s to every girl and join t to every boy. We now let every edge have unit capacity and use the Ford-Fulkerson algorithm to calculate the maximal flow. If the maximal flow value is F, then F is the maximum number of compatible marriages that can be achieved.

Here's another way to state Hall's marriage theorem.

Theorem 3.7. Let $G = G(V_1, V_2)$ be a bipartite graph. A complete matching is a subgraph of G containing every vertex such that element of V_1 is adjacent to exactly one element of V_2, and conversely. A complete matching is possible if and only if

$$|\Gamma(Y)| \geq |Y|$$

for every subset $Y \subset V_1$.

Every statement in graph theory can be reformulated in terms of properties of the adjacency matrix corresponding to the graph. To this end, we introduce the following terminology.

Definition 3.4. Let M be any $n \times n$ matrix. If $1 \leq j_1 < j_2 < \cdots < j_r \leq n$ and $1 \leq k_1 < k_2 < \cdots < k_s \leq n$, then we say that the matrix T defined by

$$T_{\ell,m} = M_{j_\ell,k_m}, \qquad 1 \leq \ell \leq r, 1 \leq m \leq s,$$

is an $r \times s$ submatrix of M. We say that it is a *zero submatrix* if its elements are all zero.

Proposition 3.8. Let $G = G(V_1, V_2)$ be a bipartite graph; the elements of V_1 (resp. V_2) will be called girls (resp. boys). Then the following statements are logically equivalent.

(i) Every r girls likes at least r boys, $1 \le r \le n$.
(ii) Let T be the $n \times n$ matrix of zeros and ones defined by $T_{jk} = 1$ if and only if girl j likes boy k. Then every $r \times s$ zero submatrix of T must satisfy $r + s \le n$.

Proof. (i) The zero submatrix with rows $j_1, \ldots, j_r$ and columns $k_1, \ldots, k_s$ corresponds to girls $g_{j_1}, \ldots, g_{j_r}$ who detest boys $g_{k_1}, \ldots, g_{k_s}$. Thus they like $n - s \ge r$ boys, so that $r + s \le n$.
(ii) If $r + s \le n$, then the r girls like $n - s \ge r$ boys.

□

Definition 3.5. An $n \times n$ matrix M is *doubly stochastic* if its elements are non-negative and every row and column sums to one. More formally, we have $M_{jk} \ge 0$, for $1 \le j, k \le n$, and

$$\sum_{\ell=1}^{n} M_{j\ell} = \sum_{m=1}^{n} M_{mk} = 1, \qquad 1 \le j, k \le n.$$

Doubly stochastic matrices arise naturally in probability theory and statistics – look up "Markov chain" in any standard text. Here, however, we shall use Hall's theorem to prove a striking and seemingly unrelated result.

Exercise 3.4. Let M be any $n \times n$ doubly stochastic matrix. Prove that

$$\sum_{j=1}^{n} \sum_{k=1}^{n} M_{jk} = n.$$

In other words, the sum of the elements of an $n \times n$ doubly stochastic matrix must equal n.

Exercise 3.5. Show that there are $n!$ different permutation matrices.

Exercise 3.6. Show that every permutation matrix is doubly stochastic.

Exercise 3.7. Show that any convex combination of doubly stochastic matrices is doubly stochastic.

Theorem 3.9. Let M be any $n \times n$ doubly stochastic matrix. Then M is a convex combination of permutation matrices. In other words, there exist permutation matrices $P_1, P_2, \ldots, P_N$ and non-negative numbers $w_1, w_2, \ldots, w_N$ satisfying $\sum_{j=1}^{N} w_j = 1$ such that

$$M = \sum_{j=1}^{N} w_j P_j.$$

To use the terminology of convex analysis, if Π denotes the set of all $n \times n$ permutation matrices and Σ denotes the set of $n \times n$ doubly stochastic matrices, the Σ is the convex hull of Π.

Proposition 3.10. *Let M be a doubly stochastic $n \times n$ matrix. Let T be the $n \times n$ matrix of zeros and ones defined by the equation $T_{jk} = 1$ if and only if $M_{jk} > 0$. (Thus T encodes the locations of the positive elements of M.) Then every $r \times s$ submatrix of T satisfies $r + s \leq n$.*

Proof. Without loss of generality, we may permute the rows and columns of M and T so that the zero submatrix appears at the top left of the matrix. Thus we obtain

$$M = \begin{pmatrix} Z & A \\ B & C \end{pmatrix},$$

where Z is an $r \times s$ matrix consisting entirely of zeros. The sum of every element of A must be equal to r, because every row of A must sum to r. Similarly, the sum of the elements of B must equal s. Since the sum of the elements of M must equal n, we deduce that $r + s \leq n$. $\square$

Now the matrix T defined in the last proposition defines a bipartite graph in the obvious way. Specifically, using the nuptial language adopted earlier, we take n girls and boys such that the jth girl likes the kth boy if and only if $T_{jk} = 1$. Thus, because every $r \times s$ submatrix of zeros satisfies $r + s \leq n$, we know that there exists a complete matching. In other words, T contains a permutation matrix, $P^{(1)}$ say.

Exercise 3.8. Show that every complete matching in T corresponds to a permutation matrix.

By definition of T, we see that

$$w_1 := \min\{M_{jk} : P^{(1)}_{jk} = 1\}$$

is a positive number. Thus the matrix

$$M_1 := \frac{M - w_1 P^{(1)}}{1 - w_1}$$

is a doubly stochastic matrix with strictly fewer positive elements than M. We can now repeat the construction, obtaining

$$\begin{aligned} M \equiv M_0 &= w_1 P^{(1)} + (1 - w_1) M_1 \qquad (3.3) \\ M_1 &= w_2 P^{(2)} + (1 - w_2) M_2 \\ M_2 &= w_3 P^{(3)} + (1 - w_3) M_3 \\ &\vdots \\ M_{N-2} &= w_{N-1} P^{(N-1)} + (1 - w_{N-1}) M_{N-1} \end{aligned}$$

$$M_{N-1} \;=\; P^{(N)},$$

where $w_1, w_2, \ldots, w_{N-1} \in [0,1]$, $M_0, M_1, M_2, \ldots, M_{N-1}$ are doubly stochastic matrices such that M_{k+1} contains strictly fewer positive elements than M_k, for $k = 0, 1, \ldots, N-2$, and $P^{(1)}, \ldots, P^{(N)}$ are permutation matrices. In other words, we have

$$M_j \in K(P^{(j+1)}, M_{j+1}), \qquad j = 0, 1, \ldots, N-2, \qquad \text{and} \qquad M_{N-1} \in K(P^{(N)}),$$

which implies

$$M = M_0 \in K(P^{(1)}, P^{(2)}, \ldots, P^{(N)}),$$

where $K(\cdot)$ denotes convex hull.

If we use the Ford-Fulkerson algorithm to construct the complete matching in the matrix T, then the above method is an efficient algorithm.

Example 3.4. The matrix

$$M \equiv M_0 = \begin{pmatrix} 2/3 & 1/3 & 0 \\ 1/3 & 0 & 2/3 \\ 0 & 2/3 & 1/3 \end{pmatrix}$$

is doubly stochastic, and we see that

$$T = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

Thus T contains the permutation matrix

$$P^{(1)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

and $w_1 = 2/3$. Therefore the matrix

$$M_1 = \frac{M_0 - w_1 P^{(1)}}{1 - w_1} = \begin{pmatrix} 0 & 1/3 & 0 \\ 1/3 & 0 & 0 \\ 0 & 0 & 1/3 \end{pmatrix}$$

is our next stochastic matrix to consider. Of course, we see immediately that $M_1 = (1/3)P^{(2)}$, where

$$P^{(2)} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Example 3.5. The matrix

$$M \equiv M_0 = \begin{pmatrix} 1/4 & 1/2 & 0 & 1/4 \\ 1/2 & 0 & 1/2 & 0 \\ 0 & 1/4 & 1/2 & 1/4 \\ 1/4 & 1/4 & 0 & 1/2 \end{pmatrix}$$

is doubly stochastic, with

$$T = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix}.$$

We see that T contains the permutation matrix

$$P^{(1)} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

and $w_1 = 1/4$. Thus our next doubly stochastic matrix is

$$M_1 = \frac{M_0 - w_1 P^{(1)}}{1 - w_1} = \frac{4}{3} \begin{pmatrix} 1/4 & 1/2 & 0 & 0 \\ 1/2 & 0 & 1/4 & 0 \\ 0 & 0 & 1/2 & 1/4 \\ 0 & 1/4 & 0 & 1/2 \end{pmatrix},$$

with corresponding matrix

$$T = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}.$$

We see that T now contains the permutation matrix

$$P^{(2)} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

and $w_2 = 2/3$. Thus our second doubly stochastic matrix is

$$M_2 = \frac{M_1 - w_1 P^{(1)}}{1 - w_1} = 3 \begin{pmatrix} 1/3 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 1/3 \\ 0 & 1/3 & 0 & 0 \end{pmatrix} \equiv P^{(3)},$$

and we recognize this as a permutation matrix. Thus we have the convex combination

$$M \quad = \quad w_1 P^{(1)} + (1 - w_1) M_1$$

$$= w_1P^{(1)} + (1-w_1)\Big(w_2P^{(2)} + (1-w_2)M_2\Big)$$
$$= w_1P^{(1)} + (1-w_1)w_2P^{(2)} + (1-w_1)(1-w_2)P^{(3)}).$$

Exercise 3.9. Why does Carathéodory's theorem imply that $N \le n^2+1$?

3.3. Rectangle Tilings

This is an interesting diversion using some simple properties of bipartite graphs. If R and $R_1, R_2, \ldots, R_m$ are rectangles in the plane, then we say that $R_1, \ldots, R_m$ *tile* R if

$$R_1 \cup R_2 \cup \cdots \cup R_m = R$$

and

$$R_j \cap R_k = \emptyset \qquad \text{for } j \ne k.$$

A classical problem of recreational mathematics is the construction of *squared squares.* Here our rectangles R and $R_1, \ldots, R_m$ must be squares with integer sides. Further, no two squares can be congruent (that is, the same squares rotated and shifted). We shall not study this beautiful problem in depth, but the ambitious reader may consult Chapter 2 of *Modern Graph Theory*, by B. Bollobas. Instead, we shall prove the following nice result.

Theorem 3.11. If the rectangles $R_1, \ldots, R_m$ each have at least one side whose length is a rational number, then the tiled rectangle R also has at least one side of rational length.

It's easy to see that, multiplying all sides by a sufficiently enormous integer, we need only address the next theorem.

Theorem 3.12. If the rectangles $R_1, \ldots, R_m$ each have at least one integer side, then the tiled rectangle R also has an integer side.

Further, without loss of generality (why?), we shall assume that

$$R = [0,a] \times [0,b] \tag{3.4}$$

and

$$R_k = [a_k + \lambda_k] \times [b_k + \mu_k], \qquad 1 \le k \le m. \tag{3.5}$$

Before embarking on the proof, we shall define a *lattice point* to be a point in $\mathbb{R}^2$ with integer coordinates. In other words, the lattice points are simply the elements of $\mathbb{Z}^2$, if the latter set is embedded in $\mathbb{R}^2$ in the obvious way.

Proof. Let L denote the set of all lattice points contained in R. Let's construct a bivariate graph whose vertices are $L \cup \{R_1, \ldots, R_m\}$ as follows: we join $(i,j) \in L$ to R_j if and only if (i,j) is a corner of R_j. Thus every rectangle is joined to 0, 2 or 4 lattice points, because at least one of its sides is an integer. In particular, our graph has an even number of edges. However, we note that $(0,0)$ is joined to exactly one rectangle. Further, every lattice point that isn't a corner of R is joined to 0, 2 or 4 rectangles. The total number of edges so far accounted for is therefore odd. Hence either one or all of the corners $(a,0)$, $(0,b)$, (a,b) must be a lattice point. □

The following alternative proof is not examinable, but is too lovely to omit. Let us define

$$f(x,y) = \cos(2\pi x)\cos(2\pi y),$$

and observe that

$$\int\int_{R_j} f(x,y)\,dx\,dy = \left(\int_{a_j}^{a_j+\lambda_j} \cos(2\pi x)\,dx\right)\left(\int_{b_j}^{b_j+\mu_j} \cos(2\pi y)\,dy\right).$$

Now at least one of λ_j, μ_j is an integer, λ_j say. Thus

$$\int_{a_j}^{a_j+\lambda_j} \cos(2\pi x)\,dx = \int_0^{\lambda_j} \cos(2\pi x)\,dx = 0,$$

which implies

$$\int\int_{R_j} f(x,y)\,dx\,dy = 0.$$

Because we have a rectangle tiling of R by $R_1,\ldots,R_m$, we obtain

$$\int\int_R f(x,y)\,dx\,dy = \sum_{j=1}^m \int\int_{R_j} f(x,y)\,dx\,dy = 0.$$

But then

$$\begin{aligned} 0 &= \int\int_R f(x,y)\,dx\,dy \\ &= \left(\int_0^a \cos(2\pi x)\,dx\right)\left(\int_0^b \cos(2\pi y)\,dy\right) \\ &= (2\pi)^{-2}\sin(2\pi a)\sin(2\pi b). \end{aligned}$$

Therefore at least one of a, b must be an integer.]

3.4. Constructing Spanning Trees

In several areas of Computer Science, it is often necessary to construct a spanning tree given a connected graph G. The details of such problems cannot be covered here, but a brief sketch may interest you. In the animation of a film such as *Toy Story*, each object is stored as a polygonal mesh in computer memory. The triangular faces of this mesh are coloured and the edges of the mesh form the topology of the object. For example, it is important that the parts of the mesh defining Buzz Lightyear's arm never be joined to those parts of the mesh defining other parts of his body, for otherwise it might appear that his arm were intermittently melting into his body. We *could* store the information by keeping the $n \times n$ adjacency matrix for each character, but this is extremely expensive. A better way is to build a tree, because we can then go from each node in the tree to its adjacent node.

It's very easy to build a tree in a connected graph. First pick any vertex and label it 1. Then add every edge leading to an adjacent vertex and label the corresponding vertices $2, 3, \ldots$. For each of the newly labelled vertices, add every edge that leads to an unlabelled adjacent vertex and label the new vertices. We continue until the tree contains every vertex of the original graph. Further, we see that the subgraph is acyclic because of the condition that edges leading to previously labelled vertices are never added to the subgraph, so denying the possibility of cycles forming. This simple algorithm is called *breadth-first search.*