Cours ENT305B
Discrete Optimisation

Wim van Ackooij

(October 30, 2023)

Contents

Chapter 1

Introduction

In this course we will investigate some basic concepts in discrete optimization. Notably we will look into so-called mixed integer linear programming. This is an approach very frequently employed in the industry. The course will address some of the basic principles underlying the algorithms and theory for these class of models.

Chapter 2

Linear Programming

2.1 Basic concepts

Definition 2.1.1. *A mapping $f : \mathbb{R}^n \to \mathbb{R}$ is called linear if for any $x, y \in \mathbb{R}^n$, $\alpha, \beta \in \mathbb{R}$, it holds that:*

$$f(\alpha x + \beta y) = \alpha f(x) + \beta f(y).$$

Remark 2.1.1. *Given a linear map $f : \mathbb{R}^n \to \mathbb{R}$, one can find $c \in \mathbb{R}^n$ such that $f(x) = \sum_{i=1}^n c_i x_i = \langle c, x \rangle = c^\mathsf{T} x$.*

Let us begin by briefly recalling some classic concepts from linear algebra that will be useful.

Definition 2.1.2. *Let us be given a $m \times n$ matrix A. The null space or kernel of A is $N(A) = \{x \in \mathbb{R}^n : Ax = 0\}$. The range of A is $R(A) = A\mathbb{R}^n = \{y \in \mathbb{R}^m : Ax = y, x \in \mathbb{R}^n\}$. The rank of a matrix $\operatorname{rank}(A)$ is the dimension of $R(A)$.*

Lemma 2.1.1. *Let us be given a $m \times n$ matrix A. The following are true:*

- *The null space of A^T is the orthogonal complement of $R(A)$, i.e., $\mathbb{R}^m = R(A) \oplus N(A^\mathsf{T})$.*
- *We have $\mathbb{R}^n = R(A^\mathsf{T}) \oplus N(A)$.*
- *The row and column rank of A are identical: $\operatorname{rank}(A) = \operatorname{rank}(A^\mathsf{T})$.*
- *The rank of A satisfies $\operatorname{rank}(A) \leq \min\{m, n\}$.*

Let us first present the standard form of such a linear program, frequently called canonical form:

$$\begin{array}{rl} \min\limits_{x\in\mathbb{R}^n} & c^{\mathsf{T}}x \\ \text{s.t.} & Ax \geq b \\ & x \geq 0, \end{array} \tag{2.1}$$

where $Ax \geq b$ is short-hand for

$$\begin{aligned} \sum_{j=1}^{n} a_{1j}x_j &\geq b_1 \\ \sum_{j=1}^{n} a_{2j}x_j &\geq b_2 \\ &\vdots \\ \sum_{j=1}^{n} a_{mj}x_j &\geq b_m. \end{aligned}$$

Linear programs can also be presented in so-called standard form:

$$\begin{array}{rl} \min\limits_{x\in\mathbb{R}^n} & c^{\mathsf{T}}x \\ \text{s.t.} & Ax = b \\ & x \geq 0. \end{array} \tag{2.2}$$

The canonical form can be cast into the standard form by introducing "slack variables", $s \geq 0$ and transforming $Ax \geq b$ into $Ax - Is = b$.

The (un-signed) linear program $\min_{x\,:Ax\geq b} c^{\mathsf{T}}x$ can also be lead back to the canonical form by writing $x = x^+ - x^-$, $x^+, x^- \geq 0$.

Exercise 2.1.1. *Make precise the details of the previous steps.*

Exercise 2.1.2. *Write the following linear program in canonical and standard form:*

$$\begin{array}{rl} \max\limits_{x} & 12x_1 + x_2 + 5x_3 \\ s.t. & x_2 - 2x_3 \geq 7 \\ & 2x_1 + x_3 \leq 10 \\ & 3x_1 - 4x_2 - 2x_3 = 3 \\ & x_1, x_3 \geq 0. \end{array}$$

Definition 2.1.3. *For a given vector $a \in \mathbb{R}^n$, $b \in \mathbb{R}$, the set*

$$H = \left\{ x \in \mathbb{R}^n \ : a^{\mathsf{T}} x = b \right\}, \tag{2.3}$$

is called a hyperplane. The set:

$$S = \left\{ x \in \mathbb{R}^n \ : a^{\mathsf{T}} x \leq b \right\}, \tag{2.4}$$

is called a half-space.

Exercise 2.1.3. *Establish that each hyperplane is a closed set. Likewise show that S is closed.*

Exercise 2.1.4. *Is the set $S = \left\{ x \in \mathbb{R}^n \ : a^{\mathsf{T}} x \geq b \right\}$ also a half-space?*

Definition 2.1.4. *A subset $C \subseteq \mathbb{R}^n$ is called convex if for any $x, y \in C$, $\lambda \in [0, 1]$, it holds $\lambda x + (1 - \lambda) y \in C$.*

Definition 2.1.5. *A set $P \subseteq \mathbb{R}^n$ is called a polyhedron if it can be described by the intersection of finitely many half-spaces. If the polyhedron is bounded, it is called a polytope.*

Lemma 2.1.2. *A set $P \subseteq \mathbb{R}^n$ is a polyhedron if and only if there exists a $m \times n$ matrix A, vector b, such that*

$$P = \left\{ x \in \mathbb{R}^n \ : Ax \leq b \right\}. \tag{2.5}$$

Exercise 2.1.5. *Prove the previous Lemma.*

Lemma 2.1.3. *Any polyhedron $P \subseteq \mathbb{R}^n$ is convex and closed.*

Exercise 2.1.6. *Prove the previous Lemma.*

Definition 2.1.6. *For a given set $S \subseteq \mathbb{R}^n$, it's convex hull is defined as the smallest convex set containing S. It is denoted as $\operatorname{Co} S$. .*

Lemma 2.1.4. *For a given set $S \subseteq \mathbb{R}^n$, it follows that*

$$\operatorname{Co} S = \left\{ x_p^{\lambda} \ : x_p^{\lambda} := \sum_{j=1}^{p} \lambda_j x_j, x_j \in S, \lambda_j \geq 0, \sum_{j=1}^{p} \lambda_j = 1, p \geq 1 \right\}.$$

Proof. We evidently have $S \subseteq \operatorname{Co} S$ and hence since $\operatorname{Co} S$ is convex, it follows that $x_2^\lambda \in \operatorname{Co} S$. We can now proceed by induction to see that any x_p^λ must also belong to $\operatorname{Co} S$. Indeed, with $\alpha = \sum_{j=1}^{p-1} \lambda_j < 1$, we get:

$$x_p^\lambda = \alpha(\sum_{j=1}^{p-1} \frac{\lambda_j}{\alpha} x_j) + (1-\alpha)x_p.$$

This yields the inclusion $\supseteq$.

Conversely the rhs is indeed convex:

$$\alpha x_p^\lambda + (1-\alpha) y_\ell^\mu = \sum_{j=1}^{p} \alpha\lambda_j x_j + \sum_{j=1}^{\ell} \mu_j (1-\alpha) y_j,$$

and this can be expressed as a $(\ell + p)$ convex combination. Since the rhs contains S and is convex, $\operatorname{Co} S$ must be contained in it. □

2.1.1 Graphically solving linear programs

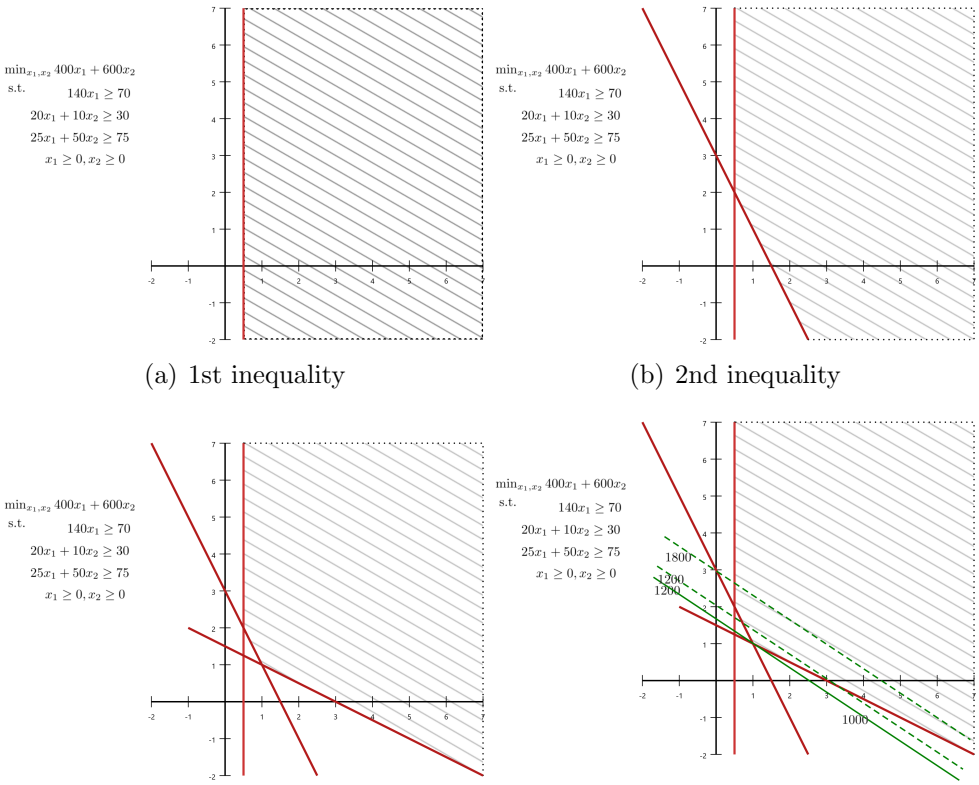
Prior to diving into more details regarding linear programming and manners in which to solve them, let us first turn our attention to the graphic solution method learned in high-school. This intuitive method will shed light on the more sophisticated situations, even if it can only be employed in dimension 2 or 3. The method consists in explicitly depicting the half spaces in order to identify the feasible set. Plotting two iso-lines of the objective function will then allow one to establish where optimality holds. The process is depicted on Figure 2.1.

Exercise 2.1.7. *Graphically solve the linear program:*

$$\begin{aligned} \max_{x_1, x_2 \geq 0} \quad & 4x_1 + 5x_2 \\ s.t. \quad & 2x_1 + x_2 \leq 800 \\ & x_1 + 2x_2 \leq 700 \\ & x_2 \leq 300. \end{aligned}$$

2.2 Geometry of polyhedra

Definition 2.2.1. *Let the convex set $C \subseteq \mathbb{R}^n$ be given. The point $x \in C$ is called a vertex (of C) if no $x_1 \neq x_2 \in C$, $\lambda \in (0,1)$ can be found such that $x = \lambda x_1 + (1-\lambda)x_2$.*



(a) 1st inequality

(b) 2nd inequality

(c) 3rd inequality

(d) Objective function

Figure 2.1: Illustration of solving a linear program graphically.

Definition 2.2.2. *For a given polyhedron $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ and $\bar{x} \in P$, the active index set $I(\bar{x})$ is given by:*

$$I(\bar{x}) = \left\{ i = 1, ..., m \ : a_i^\mathsf{T} \bar{x} = b_i \right\}. \tag{2.6}$$

Lemma 2.2.1. *Let the polyhedron $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ be given alongside $\bar{x} \in P$. There exists a neighbourhood U of $\bar{x}$, such that $I(x') \subseteq I(\bar{x})$ for all $x' \in U$.*

Proof. This results from continuity. Indeed, for $i \notin I(\bar{x})$, we have $a_i^\mathsf{T} \bar{x} < b_i$. There is thus a neighbourhood U_i of $\bar{x}$, such that $a_i^\mathsf{T} x < b_i$ for all $x \in U_i$ as well. Now $U = \bigcap_{i \notin I(\bar{x})} U_i$ and the result follows. □

Exercise 2.2.1. *Make precise the details of the previous result with the help of the ε-δ or neighbourhood definition of continuity.*

Definition 2.2.3. *For a given $m \times n$ matrix A, vector $b \in \mathbb{R}^m$ and index set $I \subseteq \{1, ..., m\}$, the restricted $|I| \times n$ matrix A_I and restricted vector $b_I \in \mathbb{R}^{|I|}$ respectively consist of forming the matrix (vector) by selecting only rows (elements) of A (b) with indexes in I.*

Proposition 2.2.1. *Let the polyhedron $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ be given alongside $\bar{x} \in P$. Then $\bar{x} \in P$ is a vertex (of P) if and only if $A_{I(\bar{x})}$ has n linearly independent rows.*

Proof. If $\bar{x} \in P$ is a vertex of P and yet $I(\bar{x})$ reduced to a singleton, we arrive at a contradiction. Indeed on a neighbourhood U of $\bar{x}$, we have $a_i^\mathsf{T} x < b_i$ for all $i \notin I(\bar{x})$. Moreover for an element in the null space of the uniquely active row y, clearly $\bar{x} \pm \delta y \in U$ for δ small enough. Now $\bar{x} = \frac{1}{2}(\bar{x} + \delta y) + \frac{1}{2}(\bar{x} - \delta y)$ leading to a contradiction with it being a vertex. In general, should $A_{I(\bar{x})}$ not have n independent rows, then by Lemma 2.1.1, the null space of A contains a non-trivial element and we can repeat the previous construction to write $\bar{x}$ as a convex combination.

Now in general, assume that $\bar{x} \in P$ is not a vertex, then we can identify x_1, x_2 and write $\bar{x} = \lambda x_1 + (1 - \lambda) x_2$ in a non-trivial way. We may assume $I(x_1) = I(x_2) = I(\mu x_1 + (1 - \mu) x_2)$ for all $\mu \in [0, 1]$. Indeed should an index $i \in I(x_1)$ not belong to $I(x_2)$, we derive $\mu a_i^\mathsf{T} x_1 + (1 - \mu) a_i^\mathsf{T} x_2 = \mu b_i + (1 - \mu) a_i^\mathsf{T} x_2 < \mu b_i + (1 - \mu) b_i = b_i$, so that i could not be active at $\bar{x}$. We now observe that for all i: $a_i^\mathsf{T}(x_2 - x_1) = 0$. But then $x_2 - x_1$ belongs to the null space of A_I. Moreover the null space of A_I is the orthogonal complement of the range of A_I^T as is known from linear algebra. But then it follows that A_I^T is not of full range, in other words it's null space must make up for

the missing dimension. We have thus identified that the rows of A_I are not linearly independent. □

Exercise 2.2.2. *Consider the polyhedron defined by $x \geq 0$, $x_2 \leq 1$, $x_1 \leq 2$, $x_1 - x_2 \geq 0$ and for a collection $\varepsilon_1, ..., \varepsilon_k \in [0,1]$: $(1-\varepsilon_j)x_1 - x_2 \geq -\varepsilon_j$. Identify the active set in the point $x = (1,1)$ belonging to P.*

Exercise 2.2.3. *By using the previous result, determine all the vertices of the following polyhedron P:*

$$\begin{aligned} 3x_1 - 2x_2 &\geq -30 \\ 2x_1 - x_2 &\geq -12 \\ x_1, x_2 &\geq 0. \end{aligned}$$

Exercise 2.2.4. *Given the polyhedron P described as follows:*

$$\begin{aligned} x_1 + 2x_2 + 2x_3 &\leq 2 \\ x_1 + 4x_2 + 2x_3 &\leq 3 \\ x_1, x_2, x_3 &\geq 0. \end{aligned}$$

Verify if the points $(0,0,0)$, $(0,0,\frac{1}{2})$ and $(0,0,1)$ are vertices of P.

Proposition 2.2.2. *Let the polyhedron $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ be given alongside $\bar{x} \in P$. Then $\bar{x} \in P$ is a vertex (of P) if and only if it uniquely solves $A_{I(\bar{x})}x = b_{I(\bar{x})}$.*

Proof. The point $\bar{x}$ is the unique solution if and only if the null space of A_I is of zero dimension if and only if A_I is of full rank. The rank of a matrix and it's transpose are identical. Hence A_I must have linearly independent rows. □

Proposition 2.2.3. *Let the polyhedron $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ be given. Then it has finitely many vertices.*

Proof. We can form the finitely many possible index sets $I \subseteq \{1, ..., m\}$, out of which we can select all those wherein A_I has n independent rows, i.e., $\operatorname{rank} A_I = n$, i.e., A_I contains an invertible block. The vertices of P are now those finitely many points that solve $A_I x = b_I$. □

Definition 2.2.4. *Let the polyhedron $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ be given. The recession cone P^∞ is defined as*

$$P^\infty = \{d \in \mathbb{R}^n : \exists t_k \downarrow 0, x_k \in P \text{ s.t. } t_k x_k \to d\}. \tag{2.7}$$

Proposition 2.2.4. *A polyhedron $P \subseteq \mathbb{R}^n$ is a polytope if and only if $P^\infty = \{0\}$.*

Proof. Pick $d \in P^\infty$, then by definition there exists $x_k \in P$, $t_k \downarrow 0$, such that $t_k x_k \to d$. Should there exist $M > 0$ such that $\|x_k\| < M$, then $\|d\| = \lim_{k\to\infty} t_k \|x_k\| = 0$. Therefore $d = 0$ and the result follows. □

Proposition 2.2.5. *The recession cone P^∞ of a polyhedron P is nontrivial if and only if there exists $x \in P$ and $d \in \mathbb{R}^n$ such that $x + \lambda d \in P$ for all $\lambda \geq 0$.*

Proof. The existence of an unbounded ray in the polyhedron clearly implies that P^∞ is not trivial.

Conversely should $0 \neq d \in P^\infty$ exist, then we can identify $x_k \in P$, $t_k \downarrow 0$, such that $t_k x_k \to d$. As already argued $\|x_k\| \to \infty$. Since however the polyhedron P can have only finitely many vertices, these are obviously bounded, and for k sufficiently large, x_k can no longer be a vertex at all. Moreover for such k large enough, we may assume that the active index set $I = I(x_k)$ remains fixed by moving to a further subsequence. Indeed should no index set repeat infinitely often, we arrive at a contradiction. We have thus found a subsequence (still denoted) x_k and fixed index set such that $A_I x_k = b_I$ and A_I is not of full rank, i.e., the null space has a non-trivial element y. Indeed, otherwise by Proposition 2.2.2 x_k would be a vertex. Now $x_k + \lambda y$, $\lambda \geq 0$ is the desired line contained in P. It is evident that $A_I(x_k + \lambda y) = b_I$, but should we not have identified the appropriate line, then for each $i \notin I$ and some λ large enough, we get $a_i^\mathsf{T}(x_k + \lambda y) > b_i$, and this would then be true for each of the base vectors of $N(A_I)$: $\{y_j\}$. However for $m > k$, $x_m - x_k \in N(A_I)$ and since $\|x_m - x_k\| \to \infty$, $x_m - x_k \neq 0$. We can now write $x_m = x_k + (x_m - x_k) = x_k + \sum_j \alpha_j y_j$. But at least one coefficient α_j must diverge and we would arrive at the contradiction that finally x_m does not belong to P. Hence the proper line was indeed identified.

□

Proposition 2.2.6. *A nonempty polyhedron P contains at least one vertex if $P^\infty = \{0\}$. .*

Proof. If the polyhedron contains no vertex, then we can not find any index set I such that the rank of A_I is n, the dimension of the space. Alternatively, the rows of A_I must be linearly dependent and this for all possible selections of index sets. As a result, for each I, the null space $N(A_I)$ must contain some non-trivial element. This would be in particular true of $I = \{1, ..., m\}$, in other words $N(A)$, but some an element belongs to P^∞. □

Exercise 2.2.5. *Show that the converse of the previous statement need not be true.*

Proposition 2.2.7. *Let the polyhedron* $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ *not be empty. Then* $P^\infty = \{d \in \mathbb{R}^n : Ad \leq 0\}$.

Proof. By definition $d \in P^\infty$ iff $x_k \in P$, $t_k \downarrow 0$ can be found such that $t_k x_k \to d$. For any fixed i, $a_i^\mathsf{T} d = \lim_{k\to\infty} t_k a_k^\mathsf{T} x_k \leq 0$.

Conversely for any d with $Ad \leq 0$ and for $x \in P$ (which exists by assumption that P is not empty), clearly $x + \lambda d \in P$ for all $\lambda \geq 0$. Moreover clearly $d = \lim_{\lambda\to\infty} \frac{1}{\lambda}(x + \lambda d)$, thus establishing $d \in P^\infty$. □

Theorem 2.2.1 (Minkowski-Weyl)**.** *Let the polyhedron* $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ *not be empty. There exists a polytope* $P' \subseteq P$ *such that* $P = P' + P^\infty$*, where the latter sum is to be understood in the Minkowski sense.*

Proof. We may assume P^∞ is not trivial. We define P' as the convex hull of the finitely many vertices. Furthermore, we have already established that for any $x \in P$, $d \in P^\infty$, $x + \lambda d \in P$, thus clearly $P' + P^\infty \subseteq P$.

The converse implication will be derived by induction on the dimension of the polyhedron. This is defined as the dimension of the affine space containing P. In fact, the latter is defined as $n - rank A^=$, with $A^=$ the matrix consisting of the index set I such that $A_I x = b_I$ for all $x \in P$. In other words, I is the largest index set contained in all active index sets.

If this dimension is 1 then P is of the form $[a, b]x_0$, $A_I x_0 = b_I$ with $a = -\infty$ and/or $b = \infty$. Regardless of this, the result follows. (N.B. : Observe that if both a, b are finite, there are two vertices, if a or b are finite there is one vertex and an element in P^∞, if both are not finite $P = P^\infty$).

We may now thus assume that for a dimension p of the Polyhedron the result holds true.

Let $x \in P$ be given but arbitrary. We may assume that $A_{I(x)}$ is not of full rank and $N(A_{I(x)})$ must contain a non-trivial element. Indeed if not, by Proposition 2.2.1 x was itself a vertex. Should however $I(x) = \emptyset$ happen, we may reason with $y \in P^\infty$ ($y \neq 0$ exists by assumption) and apply cases 1 and 2 below:

But for such a non-trivial element y, we have $A_{I(x)} y = 0$ and $a_i^\mathsf{T}(x + \varepsilon y) < b_i$ for $i \notin I(x)$ and ε small enough. Three cases can now arise:

- $y \in N(A)$, i.e., $y \in P^\infty$, in which case the polyhedron P can be written as $P'' + \alpha y$ with $y \in P^\infty$ and the dimension of P'' one less than P. We may thus leverage the

induction hypothesis in this case to establish that P can be written as indicated.

- $y \in P^\infty$ but not $y \in N(A)$, there must thus exist some $j \notin I(x)$ with $a_j^\mathsf{T} y < 0$. Hence we can find some largest $\lambda > \varepsilon$ until $x - \lambda y$ makes a new component active. We have thus generated $x' = x - \lambda y$, with $I(x') = I(x) \cup \{j\}$, $x' \in P$. If $x' \in P' + P^\infty$, we are clearly done, since $x = x' + \lambda y$ and $y \in P^\infty$.

- for all $y \in N(A_{I(x)})$, there exists some $j \notin I(x)$ with $a_j^\mathsf{T} y > 0$ and likewise there exists some $j_2 \notin I(x)$ with $a_{j_2}^\mathsf{T} y < 0$. Otherwise it would suffice to take $-y$ and apply case 2. Now we can find some $\lambda_1, \lambda_2 \geq \varepsilon$ such that both $x - \lambda_1 y$ and $x + \lambda_2 y$ have an additional active component. Since

$$x = \frac{\lambda_2}{\lambda_1 + \lambda_2}(x - \lambda_1 y) + \frac{\lambda_1}{\lambda_1 + \lambda_2}(x + \lambda_2 y),$$

  x can be written as a convex combination of these two elements. If both of these elements $(x - \lambda_1 y)$, $(x + \lambda_2 y)$ are vertices we are done. Else we may repeat the argument on either of the points.

The second or third case can only be repeated finitely often since at each time a new active index is found. The resulting index set I' must lead to $A_{I'}$ being of full rank, in other words the identification of a vertex. Remember that otherwise there exists a non trivial element in $N(A)$ and we may leverage the induction hypothesis. □

This following now immediately follows by a combination with Proposition 2.2.4:

Corollary 2.2.1. *A polytope $P \subseteq \mathbb{R}^n$ is equal to the convex hull of its finitely many vertices.*

Corollary 2.2.2 (The recession cone is a polyhedral cone)**.** *Let a polyhedron $P \subseteq \mathbb{R}^n$ be given and consider P^∞. There are finitely many bounded $y_1, ..., y_p$, such that any $y \in P^\infty$ can be written as $y = \sum_{j=1}^p \lambda_j y_j$, $\lambda_j \geq 0$.*

Proof. We may assume that P^∞ is not trivial. We can now form the polytope $P' = P^\infty \cap \{y \ : y \in [-1, 1]^n\}$. This polytope has finitely many vertices (and at least one) $y_1, ..., y_p$ and any $y \in P'$ can be written as $\sum_{j=1}^p \lambda_j y_j$ with $\lambda_j \geq 0, \sum_{j=1}^p \lambda_j = 1$. Now for any $y \in P^\infty$ we have $Ay \leq 0$ and this is the defining feature of P^∞. We have $\frac{y}{\|y\|_\infty} \in [-1, 1]^n$ and therefore: $y = \sum_{j=1}^p \|y\|_\infty \lambda_j y_j$. □

We can even use this result to establish a more powerful result:

Corollary 2.2.3. *Let $P_1, ..., P_k$ be polyhedra. Then the convex hull of the union of these polyhedra is itself a polyhedron, i.e.,* $\mathrm{Co} \bigcup_{i=1}^{k} P_i$ *is a polyhedron.*

Proof. For each $i = 1, ..., k$, we may find two finite sets of points E_i and R_i such that $P_i = \mathrm{Co}\, E_i + \mathrm{cone}\, R_i$, where cone stands for the conic convex hull. It is now our claim that we can write

$$P = \mathrm{Co} \bigcup_{i=1}^{k} P_i = \mathrm{Co} \bigcup_{i=1}^{k} E_i + \mathrm{cone} \bigcup_{i=1}^{k} R_i.$$

To this end, any $x \in P$ can be written as $x = \sum_{j=1}^{p_1} \lambda_j x_j$, with $x_j \in P_{i_j}$. But then we have

$$\begin{aligned} x &= \sum_{j=1}^{p_1} \lambda_j x_j \\ &= \sum_{j=1}^{p_1} \lambda_j (\sum_{\ell=1}^{p_2} \mu_\ell e_\ell^{i_j} + \sum_n \alpha_n r_n^{i_j}), \end{aligned}$$

which does indeed belong to the indicated decomposition. The reverse implication can be understood as follows. Any x in the rhs can be written as:

$$\begin{aligned} x &= \sum_{j=1}^{k} \sum_{e \in E_j} \lambda_{e,j} e + \sum_{r \in R_j} \mu_{r,j} r \\ &= \sum_{j=1}^{k} \sum_{e \in E_j} \lambda_{e,j} (e + \sum_{r \in R_j} \frac{\mu_{r,j}}{\lambda_{e,j}} r), \end{aligned}$$

which is a convex combination of elements in the various polyhedra P_j. Lemma 2.1.4 allows us to conclude. □

Remark 2.2.1. *The previous result is not true without the convex hull operation. Come up with a counterexample.*

2.3 Duality

The linear program (2.1) has "dual":

$$\begin{aligned} \max_{y \in \mathbb{R}^m} \quad & b^\mathsf{T} y \\ \text{s.t.} \quad & A^\mathsf{T} y \leq c \\ & y \geq 0. \end{aligned} \tag{2.8}$$

this dual can indeed be properly tied to the primal problem through Fenchel's conjugacy, but that would move us too far from the main topic of this course. Let us take the "formalism" for granted.

Exercise 2.3.1. *Write down the dual problems to the following linear program:*

$$\begin{array}{rl} \max\limits_{x} & 2x_1 + 3x_2 + 4x_3 + x_4 \\ s.t. & x_1 - 5x_3 + 2x_4 \leq 7 \\ & x_1 - 5x_2 + 2x_4 \leq 7 \\ & 2x_1 + 4x_2 - 6x_3 \leq 9 \\ & x_3 \geq 0. \end{array}$$

Proposition 2.3.1 (Weak duality)**.** *Let $\bar{x}$ be feasible for* (2.1) *and $\bar{y}$ for* (2.8)*, then it holds that:*

$$b^\mathsf{T}\bar{y} \leq c^\mathsf{T}\bar{x}. \tag{2.9}$$

Proof. From primal feasibility and $\bar{y} \geq 0$ we get $\bar{y}^\mathsf{T}(A\bar{x}-b) \geq 0$. This is $(A^\mathsf{T}\bar{y})^\mathsf{T}\bar{x} \geq b^\mathsf{T}\bar{y}$. But now combining with dual feasibility $c^\mathsf{T}\bar{x} \geq (A^\mathsf{T}\bar{y})^\mathsf{T}\bar{x}$ the result is reached. □

Corollary 2.3.1. *Assume that problem* (2.1) *admits a solution with optimal value c^*, then for any $\bar{y}$ feasible for* (2.8)*, it holds that $b^\mathsf{T}y \leq c^*$.*

Exercise 2.3.2. *Establish that the dual problem of* (2.8) *is indeed the primal problem.*

As a result of our earlier corollary we can now provide an explicit optimality certificate based on equality holding in "weak duality":

Corollary 2.3.2. *Assume given a primal feasible solution $\bar{x}$ for* (2.1) *and dual feasible solution $\bar{y}$ for* (2.8) *with $b^\mathsf{T}\bar{y} = c^\mathsf{T}\bar{x}$. Then $\bar{x}$ is primal optimal and $\bar{y}$ dual optimal.*

2.4 Fundamental theorem of linear programming

Let us now turn our attention to the important Farkas' Lemma and explain shortly after why this result is of such interest in designing algorithms. The proof is slightly involved, but the auxiliary results that we shall develop will actually also help show a powerful result of linear programming regarding strong duality. Let us first consider the following abstract looking result:

Lemma 2.4.1. *Let M be an $n \times n$ skew symmetric matrix, i.e., $M^\mathsf{T} = -M$, consider a vector $q \geq 0$ and parameter $\mu > 0$. Then the extended valued convex function $f_\mu : \mathbb{R}^n \to \bar{\mathbb{R}}$ defined as*

$$f_\mu(x) = q^\mathsf{T} x - \mu(\sum_{j=1}^{n} \log(x_j) + \sum_{j=1}^{n} \log(M_j^\mathsf{T} x + q_j), \tag{2.10}$$

with M_j denoting the jth row of M, is strictly convex, has domain

$$\mathcal{D}\mathrm{om}(f_\mu) = \prod_x \{(x, s) \ : Mx - s = -q, x > 0, s > 0\}$$

and the following are equivalent:

- *f_μ admits a (unique) minimizer*
- *There exists a solution (x, s) to*

$$Mx - s = -q, x \geq 0, s \geq 0$$
$$x_j s_j = \mu, j = 1, ..., n.$$

Proof. The statement of strict convexity of f_μ is clear as well as its domain. The latter set being open and f_μ continuously differentiable, the statement essentially follows from computing ∇f_μ and equating: $\nabla f_\mu(x) = 0$. But this is none other than

$$0 = q_i - \mu \frac{1}{x_i} - \mu (M^\mathsf{T})_i \frac{1}{s},$$

with $s = Mx + q$ and $\frac{1}{s}$ the vector with elements defined by this operation element-wise. Hence, $\mu \frac{1}{x_i} - s_i = M_i(\mu s^{-1} - x)$. This can again be rewritten as:

$$0 = \mu M_i s^{-1} - \mu x_i^{-1} s_i s_i^{-1} - M_i s^{-1} x s + x^{-1} s s^{-1} x s_i,$$

which with the matrix notation $X = \operatorname{diag} x$, $S = \operatorname{diag} s$ leads to the convenient form

$$0 = (M - X^{-1}S)S^{-1}(\mu - XS)e,$$

with e the all one vector. It is moreover known that M, as an antisymmetric matrix can only admit 0 or purely imaginary eigenvalues, therefore the matrix $M - X^{-1}S$ does not have zero eigenvalues and hence is not singular. Consequently $(M - X^{-1}S)S^{-1}$ is not singular and hence the equation can only hold true if $XSe = \mu e$, i.e., $x_i s_i = \mu$. □

Exercise 2.4.1. *Let $\mu > 0, p > 0$ be given and consider the mapping $g : \mathbb{R}_+ \to \mathbb{R}$ $x \mapsto px - \mu \log(x)$. Then show that*

- *the mapping g is bounded from below.*
- *for any κ, the set $\mathrm{lev}_\kappa\, g$ is contained in an interval of the type $[a, b]$, $0 < a < b$.*
- *the mapping g is continuous and convex.*

Finally, conclude that g has a unique minimizer.

Lemma 2.4.2. *In the setting of Lemma 2.4.1, let us assume the existence of $x_0 \in \mathcal{D}\mathrm{om}(f_\mu)$. Then, for each $\mu > 0$, f_μ admits an optimal solution $x(\mu)$ with associated variable $s(\mu) = Mx(\mu) + q$. The complementarity condition:*

$$(x(\mu) - x_0)^\mathsf{T}(s(\mu) - s_0) = 0,$$

holds and moreover, the set $\{(x(\mu), s(\mu))\}_{\bar\mu > \mu > 0}$ is bounded for all $\bar\mu < \infty$.

Proof. Observe that $-z^\mathsf{T}Mz = z^\mathsf{T}(-M)z = z^\mathsf{T}M^\mathsf{T}z = z^\mathsf{T}Mz$, so that $z^\mathsf{T}Mz = 0$ for all $z \in \mathbb{R}^n$. Hence in particular the result holds for $z = x(\mu) - x$, which gives the complementarity condition. Actually the complementarity condition holds for arbitrary $(x, s) = (x, Mx + q)$. Now (Remember $(x_0 + x)^\mathsf{T}M(x_0 + x) = 0$)

$$s_0^\mathsf{T}x + x_0^\mathsf{T}s = x^\mathsf{T}s + x_0^\mathsf{T}s_0 = q^\mathsf{T}x + q^\mathsf{T}x_0,$$

which after rearrangement allows us to write

$$f_\mu(x) = s_0^\mathsf{T}x + x_0^\mathsf{T}s - \mu(\sum_{j=1}^n \log(x_j) + \log(s_j)) - q^\mathsf{T}x_0,$$

which thus shows that minimizing f_μ amounts to minimizing $(x, s) \mapsto s_0^\mathsf{T}x + x_0^\mathsf{T}s - \mu(\sum_{j=1}^n \log(x_j) + \log(s_j))$ under the constraints $s = Mx + q$. First of all, this minimization (without constraints) amounts to minimizing $2n$ functions of the kind given in Exercise 2.4.1. The level sets of those maps where shown to be bounded (and thus compact). Hence so is the intersection with the set $s = Mx + q$, which thus shows that f_μ has bounded level sets. We may now invoke Weierstrass theorem to conclude on the existence of an optimal solution.

Now,

$$(s_0)_j x_j(\mu) \leq s_0^\mathsf{T}x(\mu) + x_0^\mathsf{T}s(\mu) = x(\mu)^\mathsf{T}s(\mu) + x_0^\mathsf{T}s_0 = n\mu + x_0^\mathsf{T}s_0 \leq n\bar\mu + x_0^\mathsf{T}s_0,$$

where $x(\mu)^\mathsf{T}s(\mu) = n\mu$ follows from Lemma 2.4.1. Hence, $x_j(\mu) \in [0, \frac{\bar\mu + x_0^\mathsf{T}s_0}{(s_0)_j}]$ and the $s_j(\mu) \in [0, \frac{\bar\mu + x_0^\mathsf{T}s_0}{(x_0)_j}]$ from a similar argument. □

However just prior to establishing the Farkas' Lemma, we require an abstract result showing that certain problems admit "strict complementarity".

Lemma 2.4.3. *Consider the linear program*

$$\min_{z\in\mathbb{R}^n} 0^{\mathsf{T}} z,\ s.t. Mz \geq 0, z \geq 0,$$

where M is skew symmetric, i.e., $M^{\mathsf{T}} = -M$. Then this linear program is self-dual and feasible and there exists an optimal solution $\bar{z}$ satisfying the conditions:

$$\bar{z} \geq 0, M\bar{z} \geq 0 \text{ and } \bar{z} + M\bar{z} > 0. \tag{2.11}$$

Proof. The dual is indeed nothing other than the problem itself, as follows from immediate identification. Let us first observe that $\bar{z} = 0$ is always optimal, so that such optimal solutions exist. Now let $\bar{z}$ be an arbitrary feasible (and thus optimal solution), then clearly $\bar{z} \geq 0$, $M\bar{z} \geq 0$ and thus $M\bar{z} + \bar{z} \geq 0$. Now observe that $-z^{\mathsf{T}}Mz = z^{\mathsf{T}}(-M)z = z^{\mathsf{T}}M^{\mathsf{T}}z = z^{\mathsf{T}}Mz$, so that $z^{\mathsf{T}}Mz = 0$ for all $z \in \mathbb{R}^n$.

For the remaining claim, let us define the following, $r = e - Me$, with e the all one vector, $q = (0, n+1)$ and

$$\bar{M} = \begin{bmatrix} M & r \\ -r^{\mathsf{T}} & 0 \end{bmatrix},$$

then $\bar{M}$ is still skew-symmetric and evidently $\bar{M}(e,1) + q = (e,1)$. We may thus set $\bar{x}_0 = (e,1)$ and $\bar{s}_0 = (e,1)$, to entail the existence of an optimal sequence $(\bar{x}(\mu), \bar{s}(\mu))$ for minimizing f_μ defined with these terms. Since, Lemma 2.4.2 furthermore tells us that this sequence is actually bounded, we may upon passing to the limit $\mu \to 0$, entail the existence of a cluster point $(\bar{x}^*, \bar{s}^*)$. The latter satisfies $(\bar{x}^*)^{\mathsf{T}}\bar{s}^* = 0$, since $\bar{x}(\mu)^{\mathsf{T}}s(\mu) = (n+1)\mu$. Obviously, $\bar{s}^* = \bar{M}\bar{x}^*$ by continuity and now a computation shows that

$$(\bar{x}^*)^{\mathsf{T}}\bar{s}^* = x^*Mx^* + ((x^*)^{\mathsf{T}}r)x^*_{n+1} - x^*_{n+1}(r^{\mathsf{T}}x^*) + (n+1)x^*_{n+1} = (n+1)x^*_{n+1},$$

where $\bar{x}^* = (x^*, x^*_{n+1})$. So $x^*_{n+1} = 0$ must hold true and hence $s^* = Mx^* \geq 0$, $x^* \geq 0$ is indeed feasible and optimal for the optimization problem at hand.

Using the complementarity condition of Lemma 2.4.2 (with (z^*, Mz^*) replacing (x_0, s_0)),

gives

$$\begin{aligned} n\mu &= (z^*)^\mathsf{T} s(\mu) + x(\mu)^\mathsf{T}(Mz^*) \\ &= \sum_{j\,:z_j^*>0} z_j^* s_j(\mu) + \sum_{j\,:M_j z^*>0} x_j(\mu) M_j z^*, \\ n &= \sum_{j\,:z_j^*>0} \frac{z_j^*}{x_j(\mu)} + \sum_{j\,:M_j z^*>0} \frac{M_j z^*}{s_j(\mu)}, \end{aligned}$$

where in the last equation we have divided the first by μ and recalled that $\mu = x_j(\mu)s_j(\mu)$. Taking the limit with respect to $\mu \to 0$, we find that $\left|j\ : z_j^* > 0\right| + |j\ : M_j z^* > 0| = n$.

From feasibility of z, it is clear that $z + Mz \geq 0$ for all feasible solutions, hence in particular for z^*. Assume now that we may find some i such that $z_i + M_i z = 0$, then $z_i = 0$ and $M_i z = 0$. Moreover $z^\mathsf{T} M z = 0$, shows that not both can be strictly positive, hence $\{j = 1, ..., n\ : z_j > 0\}$ and $\{j = 1, ..., n\ M_i z > 0\}$ are disjoint. Hence under the assumption that $z_i^* + M_i z^* = 0$ for some i, $\left|j\ : z_j^* > 0\right| + |j\ : M_j z^* > 0| \leq n-1$, which is a contradiction.

□

Lemma 2.4.4 (Farkas)**.** *For a given $m \times n$ matrix A and vector $b \in \mathbb{R}^m$, either there exists a solution x such that*

$$Ax \geq b,\ x \geq 0, \tag{2.12}$$

or there exists a solution y

$$A^\mathsf{T} y \leq 0,\ b^\mathsf{T} y > 0,\ y \geq 0, \tag{2.13}$$

but both can not hold true simultaneously.

Proof. Should both systems admit a solution pair $(\bar{x}, \bar{y})$, then

$$0 < b^\mathsf{T}\bar{y} \leq (A\bar{x})^\mathsf{T}\bar{y} = (\bar{x})^\mathsf{T} A^\mathsf{T}\bar{y} \leq 0,$$

which is a contradiction.

Define the matrix M as follows,

$$M = \begin{bmatrix} 0 & A & -b \\ -A^\mathsf{T} & 0 & 0 \\ b^\mathsf{T} & 0 & 0 \end{bmatrix},$$

and $z = (y, x, \kappa)$. This matrix is skew-symmetric and we may apply Lemma 2.4.3 to identify a strictly complementary solution z, with $s(z) = (Ax - \kappa b, -A^\mathsf{T} y, b^\mathsf{T} y)$. In

particular therefore $\kappa \geq 0$ and should we look at $\kappa = 0$, the existence of a solution to the second series of equations follows. The case $\kappa > 0$ can well be normalized to $\kappa = 1$, since whenever z is an optimal solution to the problem, then so is λz for $\lambda > 0$, moreover, clearly $\lambda z + M\lambda z = \lambda(z + Mz) > 0$, too. This thus leads to the first series of equations holding.

□

A consequence of Farkas' Lemma is the fundamental result of linear programming:

Theorem 2.4.1. *Consider the linear program* (2.1) *and its dual* (2.8)*. Then either one of the following alternatives can occur:*

- *Both the primal and dual program admit an optimal solution and strong duality holds. The optimal solution pair can moreover be taken strictly complementary.*
- *Both the primal and dual have empty feasible set.*
- *The primal problem is infeasible and the dual program unbounded.*
- *The dual program is infeasible and the primal problem unbounded.*

Proof. The third and fourth item are mutually exclusive due to Farkas' Lemma 2.4.4, once the second has been accounted for. Indeed, if the primal problem is infeasible, there exists $y \geq 0$, $A^\mathsf{T} y \leq 0$, $b^\mathsf{T} y > 0$ by Farkas' Lemma. Furthermore, there exists at least one $\bar{y} \geq 0$ such that $A^\mathsf{T}\bar{y} \leq c$. Now for any $\lambda > 0$, $\bar{y} + \lambda y$ is dual feasible, thus showing that the dual is indeed unbounded. The proof of the fourth item is similar. As for the first item, let us consider the extended matrix:

$$M = \begin{bmatrix} 0 & A & -b \\ -A^\mathsf{T} & 0 & c \\ b^\mathsf{T} & -c^\mathsf{T} & 0 \end{bmatrix},$$

which once more is skew-symmetric. We can thus find a vector $z = (y, x, \kappa)$, such that $z \geq 0$, $Mz \geq 0$ and $z + Mz > 0$ as a result of Lemma 2.4.3. Observe that strict complementarity yields $\kappa > 0$ or $\kappa = 0$. In the first case, $b^\mathsf{T} y - c^\mathsf{T} x = 0$ must hold true, and we have thus identified a primal-dual strictly complementary solution. In the second case $b^\mathsf{T} y - c^\mathsf{T} x > 0$. But since clearly λz for any $\lambda > 0$ is also feasible and also satisfies the conditions, we may find for all $C > 0$, strictly complementary x, y with $b^\mathsf{T} y - c^\mathsf{T} x > C$, so that either the third or fourth case hold true (either the primal or dual are unbounded). □

Example 2.4.1. *As a concrete example of a linear programming problem pair both of which have empty feasible set consider the data:*

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

$c = (-1, -1)$, $b = (0, 1)$. *The primal problem is infeasible due to* $0x_1 + 0x_2 \geq 1$ *and the dual as a result of the constraint* $1 \leq y_1 \leq -1$.

2.5 Optimality and strong duality

Theorem 2.5.1 (Strong duality)**.** *Consider the primal-dual pair of linear programs* (2.1) *and* (2.8) *and assume that at least one has non-empty feasible set. Then it holds that the optimal values of both problems are identical. Here we have taken the convention that the value of an infeasible minimization problem is* ∞ *and that of a maximization problem* $-\infty$.

If at least one problem admits an optimal solution, then we can find $(\bar{x}, \bar{y})$ *primal dual feasible such that* $b^\mathsf{T}\bar{y} = c^\mathsf{T}\bar{x}$. *Consequently* $\bar{x}$ *is primal optimal,* $\bar{y}$ *is dual optimal.*

Proof. This follows immediately from the fundamental Theorem of Linear Programming, i.e., Theorem 2.4.1. □

We can make explicit the convex normal cone to polyhedra:

Proposition 2.5.1. *Let the polyhedron* $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ *be given. Then the convex normal cone to* P *satisfies:*

$$\mathrm{N}_P(\bar{x}) = A^\mathsf{T}\,\mathrm{N}_{(-\infty,0]^m}(A\bar{x} - b)$$

Proof. Any $\lambda \in \mathrm{N}_{(-\infty,0]^m}(A\bar{x} - b)$ is such that $\lambda \geq 0$, $\lambda_i = 0$ if $(A\bar{x} - b)_i < 0$. Elements in the rhs are thus of the form $A^\mathsf{T}_{I(\bar{x})}\mu$, $\mu \geq 0$, with λ some extension of μ by padding it with zeros.

Now for any $x \in P$, $\left\langle A^\mathsf{T}_{I(\bar{x})}\mu, x - \bar{x} \right\rangle = \left\langle \mu, A_{I(\bar{x})}x - b_{I(\bar{x})} \right\rangle \leq 0$, thus ensuring that the right-hand side belongs to $\mathrm{N}_P(\bar{x})$.

Conversely, let us assume the existence of some $\lambda \in \mathrm{N}_P(\bar{x})$ that can not be written as $\lambda = A^\mathsf{T}_I\mu$ with $\mu \geq 0$ and $I := I(\bar{x})$. Applying Farkas' Lemma 2.4.4, yields the existence of some $y \neq 0$ such that $A_I y \leq 0$, $\lambda^\mathsf{T} y > 0$. Now as a result for some $\varepsilon > 0$ small

enough, $\bar{x}+\varepsilon y \in P$, since we can identify some neighbourhood U of $\bar{x}$ with $I(x) \subseteq I(\bar{x})$ for $x \in U$. Moreover $A_I(\bar{x}+\varepsilon y) \leq b_I$. As a consequence $\langle \lambda, \bar{x}+\varepsilon y - \bar{x}\rangle = \varepsilon \langle \lambda, y\rangle > 0$, but this contradicts $\lambda \in \mathrm{N}_P(\bar{x})$. □

Theorem 2.5.2 (Optimality conditions)**.** *The linear program* (2.1) *has optimal solution* $\bar{x}$ *if and only if* $(\bar{x}, \bar{y})$ *solves:*

$$\begin{aligned} 0 &= c - A^{\mathsf{T}} y - \mu \\ b &\leq Ax \\ 0 &= y^{\mathsf{T}}(b - Ax) \\ 0 &\leq y, x \geq 0, \mu \geq 0 \\ 0 &= \mu^{\mathsf{T}} x \end{aligned}$$

Exercise 2.5.1. *Establish the previous result formally by using the abstract optimality conditions* $\min_{x\in C} f(x) : 0 = \nabla f(x) + \mathrm{N}_C(x)$*, with* f C^1*,* C *convex closed.*

2.6 Structure of solutions

Let us now discuss some facts about characterizing the optimal solutions of linear optimization problems.

Lemma 2.6.1. *Consider the linear program* (2.1) *and assume that it admits an optimal solution. Then there exists also a "basic" optimal solution, i.e., with* I *the active index set of* $Ax \geq b$ *at this solution,* A_I*,* b_I *the restriction of* A *and* b *to the set of active rows (which we may assume linearly independent), we may find a sub-selection of columns of* A_I*, leading to a matrix* $\tilde{A}_I$*, which is invertible such that* $\tilde{x} = (\tilde{A}_I)^{-1} b_I$ *is also optimal.*

Proof. Let us write the feasible set $C = \{x \ : Ax \geq b, x \geq 0\}$, or alternatively with $\tilde{A} = [A, I]$ and $\tilde{b} = [b, 0]$ as $C = \left\{x \ : -\tilde{A}x \leq -\tilde{b}\right\}$ and assume that x^* is the identified optimal solution.

The first order optimality conditions now give us $0 \in c + \mathrm{N}_C(x^*)$. The calculus rules of normal cones given in Proposition 2.5.1 allow us to express these optimality conditions in the following explicit form:

$$0 = c - \lambda^{\mathsf{T}} \tilde{A},$$

with $\lambda_i = 0$ when $\tilde{A}_i^{\mathsf{T}} x^* > \tilde{b}_i$ and $\lambda_i \geq 0$ otherwise. With I denoting the active index set, and recalling Theorem 2.4.1 and the strict complementarity Lemma 2.4.3, we may assume that $\lambda_I > 0$ holds true (where the latter denotes the restriction to I of λ as above).

With $J = \{i = 1, ..., n \ : x_i^* > 0\}$, assume the existence of some index $j \in J$ and appropriate multipliers μ_i such that $\tilde{A}_j = \sum_{i \in J, i \neq j} \mu_i \tilde{A}_i$, i.e., column j is linearly dependent of other (active) columns. Then, let us define the new solution:

$$x_i = x_i^* + \mu_i x_j^*, i \in J \setminus \{j\},$$

and set to zero elsewhere. Then,

$$\begin{aligned}\sum_{i=1}^{n} \tilde{A}_i x_i = \sum_{i \in J} \tilde{A}_i x_i &= \sum_{i \in J, i \neq j} \tilde{A}_i x_i^* + (\sum_{i \in J, i \neq j} \tilde{A}_i \mu_i) x_j^* \\ &= \sum_{i \in J, i \neq j} \tilde{A}_i x_i^* + \tilde{A}_j x_j^* = \sum_{i \in J} \tilde{A}_i x_i^* = \sum_{i=1}^{n} \tilde{A}_i x_i^* \leq \tilde{b},\end{aligned}$$

so that x is also a feasible solution. Moreover, recalling that for $i = 1, ..., n$, $c_i = \lambda^\mathsf{T} \tilde{A}_i$,

$$\begin{aligned}\sum_{i=1}^{n} c_i x_i &= \sum_{i \in J, i \neq j} c_i x_i^* + (\sum_{i \in J, i \neq j} \mu_i c_i) x_j^* \\ &= \sum_{i \in J, i \neq j} c_i x_i^* + (\sum_{i \in J, i \neq j} \mu_i \lambda^\mathsf{T} \tilde{A}_i) x_j^* \\ &= \sum_{i \in J, i \neq j} c_i x_i^* + \lambda^\mathsf{T} (\sum_{i \in J, i \neq j} \mu_i \tilde{A}_i) x_j^* \\ &= \sum_{i \in J, i \neq j} c_i x_i^* + \lambda^\mathsf{T} \tilde{A}_j x_j^* = \sum_{i \in J, i \neq j} c_i x_i^* + c_j x_j^* = \sum_{i=1}^{n} c_i x_i^*.\end{aligned}$$

Hence x has an objective function value as good as x^* and must thus also be optimal. Moreover x has one linearly dependent columns less than x^*. By induction we may thus proceed in this fashion until we have produced a solution with active columns that are linearly independent. Observing that if (2.1) admits an optimal solution then so does (2.8) and in this case λ_I above is actually the dual optimal solution, we may likewise eliminate rows (from I) until the rows are also linearly independent. The resulting sub-matrix $\tilde{A}_I$ must then be invertible as claimed and with $\tilde{A}_I x = b_I$ holding, the result follows. □

The following lemma starts by providing some further insights into the structure of optimal solutions to linear problems:

Lemma 2.6.2. *Consider the linear problem* (2.1) *and assume that it admits an optimal solution. Then either one of the following cases occurs:*

- *The cost vector $c = 0$ and all feasible solutions are optimal. This is moreover the only situation wherein strictly feasible points are optimal.*
- *A single constraint is active, and if this happens, c is a multiple of that active row (or of some unit vector). The converse is also true if the constraint is not redundant.*
- *At least two distinct constraints are active, and this happens if and only if the optimal solution is an extremal point of the feasible set.*

Proof. Let us write the feasible set $C = \{x : Ax \geq b, x \geq 0\}$, or alternatively with $\tilde{A} = [A, I]$ and $\tilde{b} = [b, 0]$ as $C = \left\{x : -\tilde{A}x \leq -\tilde{b}\right\}$ and assume that x^* is the (a) optimal solution, which we may assume "basic" as a result of Lemma 2.6.1 above.

The first order optimality conditions now give us $0 \in c + \mathrm{N}_C(x^*)$. The calculus rules of normal cones given in Proposition 2.5.1 allow us to express these optimality conditions in the following explicit form:

$$0 = c - \lambda^{\mathsf{T}}\tilde{A},$$

with $\lambda_i = 0$ when $(\tilde{A})_i^{\mathsf{T}} x^* > \tilde{b}_i$ and $\lambda_i \geq 0$ otherwise.

When $c = 0$, clearly any feasible solution is optimal. Conversely when no constraints are active at x^*, then $\lambda = 0$ is the only possible dual vector and hence $c = 0$ can be concluded from the above given first order optimality conditions.

Should now a single constraint be active, then for some index i, $\lambda_i > 0$ can happen. Except for this single element, the vector λ is the zero vector. Should $\lambda_i = 0$ be possible, then once more we can conclude that $c = 0$. Otherwise $\lambda_i > 0$ must occur and $c = \lambda_i \tilde{A}_i$, which concretely means that either c is a positive multiple of the ith unit vector or of some row of A.

In the last case, at least two constraints must be active. Let I be the index set of active constraints. Now should there be an index $i \in I$ for which $\lambda_i = 0$ is possible, then on the one hand we have $c = \sum_{j \in I, j \neq i} \lambda_j \tilde{A}_j$, but also $c = \sum_{j \in I} \tilde{\lambda}_j \tilde{A}_j$ for potentially another set of multipliers with $\tilde{\lambda}_i > 0$. From this we can derive $\tilde{A}_i = \tilde{\lambda}_i^{-1} \sum_{j \in I, j \neq i} (\lambda_j - \tilde{\lambda}_j)\tilde{A}_j$, thus showing that in this case $\tilde{A}_i$ is a linear combination of other rows of $\tilde{A}$. Should I consist of only two indices, then either $\tilde{A}_i = 0$, in which case from constraint i being active we deduce $\tilde{b}_i = 0$, which can reasonably be excluded (why would one add the constraint $0 \leq 0$ to a model) or $\tilde{A}_i \neq 0$. In the last case, $\lambda_j \neq \tilde{\lambda}_j$ and $\tilde{A}_i = \mu \tilde{A}_j$, with $\mu = \tilde{\lambda}_i^{-1}(\lambda_j - \tilde{\lambda}_j)$. Since both constraints are active, we deduce:

$$b_i = (\tilde{A}_i)^{\mathsf{T}} x^* = \mu (\tilde{A}_j)^{\mathsf{T}} x^* = \mu b_j,$$

so that the constraint $\tilde{A}_i \geq \tilde{b}_i$ and $\tilde{A}_j \geq \tilde{b}_j$ are not distinct.

Should x^* now be not extremal and hence written as $\lambda x_1 + (1-\lambda)x_2$ for two feasible solutions x_1, x_2 distinct from x^*. Then should $c^\mathsf{T} x_1 > c^\mathsf{T} x^*$ hold, it would follows that $c^\mathsf{T} x^* = \lambda c^\mathsf{T} x_1 + (1-\lambda) c^\mathsf{T} x_2 > c^\mathsf{T} x^*$ which is a contradiction. Hence $c^\mathsf{T} x^* = c^\mathsf{T} x_1 = c^\mathsf{T} x_2$ and in particular any vector on the line $[x_1, x_2]$ is optimal. We may now argue that near x^*, by continuity, the active index set I can not increase. Hence let us parametrize $[x_1, x_2]$ as $x(\alpha) = \alpha x_1 + (1-\alpha)x_2$ and define $I(\alpha)$ as the active index set at α. Then for α sufficiently close to λ, we have $I(\alpha) \subseteq I$. If $I(\alpha)$ contains only a single index i for some α, then for some $\mu > 0$ it follows $c = \mu \tilde{A}_i$, so that for all α, only the constraint i can be made active. This thus contradicts the assumption at x^*. Else, each $I(\alpha)$ must contain at least two indices for all α. Arguing as above, through convexity, actually any index $i \in I$, but $i \notin I(0)$, leads to the contradiction:

$$b_i = \tilde{A}_i^\mathsf{T} x(\lambda) = \lambda \tilde{A}_i^\mathsf{T} x_1 + (1-\lambda)\lambda \tilde{A}_i^\mathsf{T} x_2 < b_i,$$

since $i \notin I(0)$. The argument is similar for $I(1)$ and so we establish that $I = I(\alpha)$ for all α. But then we have established the existence of multiple solutions to $A_I x = b_I$, which implies that columns of A_I must be linearly dependent, but this is ruled out by Lemma 2.6.1 and x^* being basic. □

2.7 Solving Linear programs with the Simplex Method

We have deduced from Lemma 2.6.2 that all solutions of a linear program are vertexes of the feasible polyhedron. Moreover these vertices are finite in number. A naive solution approach would thus simply consist in enumerating all possible vertices, computing the corresponding objective function value and find the best solution. Of course in practice it will be inefficient to generate all vertices from the start. A better approach consists however in the so-called "simplex method" which guides us efficiently through the list of candidates.

The simplex method starts with the linear program in standard form (2.2). We will partition the index set $\{1, ..., n\}$ in $B \cup N$, with B the set of base-variables, N the set of non-base variables. The set B is assumed to be so that A^B, the matrix consisting of the selection of columns of A with indexes in B is not-singular.

Remark 2.7.1. *With a linear program in standard form resulting from the canonical form having $b \leq 0$, the identification of a base B is simple. It suffices to pick all indexes belonging the slack variables in the base. This is the cheapest possible situation.*

Likewise any vector will be partitioned into x_B, x_N accordingly. As a result $Ax = A^B x_B + A^N x_N$.

Definition 2.7.1. *Let B, N be a given basis, non-basis partition of $\{1, ..., n\}$.*
A given vector $\bar{x}$ is called a basic solution of the system if $\bar{x}_B = (A^B)^{-1}b$ and $\bar{x}_N = 0$.
The basic solution is called basic feasible if moreover $\bar{x}_B \geq 0$ holds true.

As observed if we derived the standard form from a canonical form, the identification of a first feasible base is simple. Otherwise we will have to pick some arbitrary base allowing the corresponding solution vector potentially to not be feasible. The simplex method can be preceded by a so called first phase leading up to a first feasible base. It will be easier to present that phase, once the general principle is laid down. So to begin we will assume available a feasible point: a feasible start. Obtaining a feasible starting point is a relevant question for most if not all algorithms.

Exercise 2.7.1. *Given the linear system:*

$$\begin{aligned} 2x_1 - 3x_2 + 4x_3 + 2x_4 + 5x_5 + x_6 &= 7 \\ x_1 + 2x_2 + x_3 + 4x_4 + 2x_5 + 2x_6 &= 8 \\ -x_1 + 4x_2 + 4x_3 + 3x_4 - x_5 + x_6 &= 2 \\ x_1, ..., x_6 &\geq 0. \end{aligned}$$

Verify if the following matrices are basis matrices of A and if they are feasible:

$$B_1 = \begin{bmatrix} 1 & 2 & 2 \\ 2 & 1 & 4 \\ 1 & -1 & 3 \end{bmatrix}, B_2 = \begin{bmatrix} -3 & 5 & 2 \\ 2 & 2 & 4 \\ 4 & -1 & 3 \end{bmatrix}, B_3 = \begin{bmatrix} 2 & -3 & 1 \\ 1 & 2 & 2 \\ -1 & 4 & 1 \end{bmatrix}$$

corresponding to $B = \{6, 1, 4\}$, $B = \{2, 1, 4\}$ and $B = \{1, 2, 6\}$ respectively.

2.7.1 Feasible start

The following result is a restatement of Proposition 2.2.2:

Corollary 2.7.1. *Let $P = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\}$ be a polyhedron (in standard form). Then $\bar{x} \in P$ is a vertex if and only if $\bar{x}$ is a basic feasible solution.*

Any given feasible solution x can be written as $b = Ax = A^B x_B + A^N x_N$. We can premultiply this equality with $(A^B)^{-1}$ to obtain:

$$(A^B)^{-1}b = Ix_B + (A^B)^{-1}A^N x_N.$$

From this we can deduce that:

$$\begin{aligned} c^{\mathsf{T}}x &= c_B^{\mathsf{T}}x_B + c_N^{\mathsf{T}}x_N \\ &= c_B^{\mathsf{T}}((A^B)^{-1}b - (A^B)^{-1}A^N x_N) + c_N^{\mathsf{T}}x_N \\ &= (c_N - ((A^B)^{-1}A^N)^{\mathsf{T}}c_B)^{\mathsf{T}}x_N + c_B^{\mathsf{T}}(A^B)^{-1}b. \end{aligned}$$

Now $c_B^{\mathsf{T}}(A^B)^{-1}b$ is exactly the objective function value of a basic solution $\bar{x}_B$. We will give the partial cost vector a name:

Definition 2.7.2. *Let B, N be a given basis, non-basis partition of $\{1, ..., n\}$. The partitioned cost vector*

$$(0, c_N - ((A^B)^{-1}A^N)^{\mathsf{T}}c_B)^{\mathsf{T}}(x_B, x_N) = c^{\mathsf{T}}x,$$

is called reduced costs.

With these elements we can now form the Simplex Tableaux:

	Basic variables	Non basic variables	
	x_1 ... x_m	x_{m+1} ... x_n	
x_1 $\vdots$ x_m	I	$(A^B)^{-1}A^N$	$(A^B)^{-1}b$
	0	$c_N - ((A^B)^{-1}A^N)^{\mathsf{T}}c_B$	$-c_B^{\mathsf{T}}(A^B)^{-1}b$
	Reduced cost		- basic value

Table 2.1: General form of Simplex Tableau.

Example 2.7.1. *Let us return to the example solved graphically. Let us pick as a starting base $B = \{x_1, x_2, s_3\}$, then the simplex tableau becomes as in Table 2.2*

	x_1	x_2	s_3	s_1	s_2	
x_1	1	0	0	$-\frac{1}{140}$	0	$\frac{1}{2}$
x_2	0	1	0	$\frac{2}{140}$	$-\frac{1}{10}$	2
s_3	0	0	1	$\frac{75}{140}$	-5	$\frac{75}{2}$
	0	0	0	$-\frac{800}{140}$	60	-1400

Table 2.2: Simplex Tableau of graphical example

Ofcourse our reorganisation of the simplex tableau regrouping the identity matrix in the left half is just "notational trick". In reality we would not bother to permutate

any columns and rather keep the column indexes simple in the order of the variables. Knowing the basis variables then simply amounts to finding the appropriate identity vectors.

Theorem 2.7.1. *Let B, N be a given basis, non-basis partition of $\{1, ..., n\}$ with feasible basic solution $\bar{x}$. Then $\bar{x}$ is optimal for* (2.2) *if and only if the reduced cost vector satisfies $c_N - ((A^B)^{-1}A^N)c_B \geq 0$.*

Proof. With $\gamma = c_N - ((A^B)^{-1}A^N)c_B$, any feasible x has objective function value $c^\mathsf{T}x = \gamma^\mathsf{T}x_N + c^\mathsf{T}\bar{x}$. In other words $\gamma^\mathsf{T}x_N = c^\mathsf{T}(x - \bar{x})$. With $x_N \geq 0$, $\gamma \geq 0$ or $c^\mathsf{T}x \geq c^\mathsf{T}\bar{x}$ the result follows. □

Example 2.7.2. *The simplex tableau from our graphical example immediately shows that the current basic solution is not optimal.*

Proposition 2.7.1. *Let B, N be a given basis, non-basis partition of $\{1, ..., n\}$ with feasible basic solution $\bar{x}$. If there exists a non-basis solution having negative reduced cost and only non-positive elements in the simplex tableau, then problem* (2.2) *is unbounded.*

Proof. Let us call that non-basic variable q and let N' be the non-basis set without q. Then we have: $b = x_B + A^{N'}x_{N'} + A^q x_q$ and we also have $c^\mathsf{T}x = c^\mathsf{T}\bar{x} + \gamma_{N'}^\mathsf{T}x_{N'} + \gamma_q x_q$, with $\gamma_q < 0$. Now we can write $x_B = b - A^{N'}x_{N'} - A^q x_q$ and x_q can become arbitrarily large without making any x_B zero. Doing so will send the cost function to $-\infty$. □

The main idea of the Simplex method is now to effectuate so-called pivot operations bringing non-basis elements into the basis while having basis elements leave the base.

Definition 2.7.3. *A simplex pivot consists of a pair $i \in N$, $j \in B$, with reduced cost $\gamma_i < 0$ and $((A^B)^{-1}A^N)_{ji} > 0$. A simplex pivot operation consists in the matrix operations effectively ensuring that i enters the base.*

Lemma 2.7.1 (Feasible pivots)**.** *Let B, N be a given basis, non-basis partition of $\{1, ..., n\}$ with feasible basic solution $\bar{x}$. Let a simplex pivot (i, j) be given. When $j \in B$ is selected as :*

$$j \in \underset{k=1,...,m,((A^B)^{-1}A^N)_{ki}>0}{\operatorname{argmin}} \frac{(A^B)^{-1}b_k}{((A^B)^{-1}A^N)_{ki}}, \tag{2.14}$$

then the new basis $B^+ := B \cup \{i\} \setminus j$, $N^+ := N \cup \{j\} \setminus i$ is also feasible.

Proof. Indeed $x_i = \frac{\bar{x}_j}{((A^B)^{-1}A^N)_{ji}} \geq 0$ and for any index $k \in B^+$, $k \neq i$ we have:

$$x_k = \bar{x}_k - ((A^B)^{-1}A^N)_{ki}\frac{\bar{x}_j}{((A^B)^{-1}A^N)_{ji}},$$

and $\frac{\bar{x}_j}{((A^B)^{-1}A^N)_{ji}} \leq \frac{\bar{x}_k}{((A^B)^{-1}A^N)_{ki}}$ by definition of j so that $x_k \geq 0$ follows. □

Definition 2.7.4. *Let B, N be a given basis, non-basis partition of $\{1, ..., n\}$ with feasible basic solution $\bar{x}$. The solution $\bar{x}$ is called degenerate if there exists $i \in B$ with $x_i = 0$. A pivot (i, j) is called degenerate if $\bar{x}_j = 0$, i.e., if j corresponds to a degenerate element.*

Example 2.7.3. *Let us return to the example solved graphically and our earlier simplex tableau. Let us pick (s_1, s_3) as a pivot (this is the only feasible pivot). Then the simplex tableau becomes:*

The last Tableau shows optimality of $x = (1, 1)$ with optimal value 1000.

	x_1	x_2	s_3	s_1	s_2	
x_1	1	0	$\frac{1}{75}$	0	$-\frac{1}{15}$	1
x_2	0	1	$-\frac{2}{75}$	0	$\frac{1}{30}$	1
s_3	0	0	$\frac{140}{75}$	1	$-\frac{700}{75}$	70
	0	0	$\frac{800}{75}$	0	$\frac{500}{75}$	-1000

One of the difficulties in the simplex method lies in properly choosing the pivot element. Such a strategy is called a pivot rule. The simplest rule is Bland's rule. It consists of selecting the smallest indexes at each possible choice.

Definition 2.7.5 (Bland's pivot rule)**.** *Bland's pivot rule consists in selecting the smallest index i with negative reduced cost and the smallest index j ensuring a feasible pivot.*

Lemma 2.7.2. *Let B, N be a given basis, non-basis partition of $\{1, ..., n\}$ with feasible basic solution $\bar{x}$. After a non-degenerate feasible pivot operation (i, j), the value of the new basis variable is*

$$x_i = \frac{((A^B)^{-1}b)_j}{((A^B)^{-1}A^N)_{ji}} > 0,$$

and the new cost is $c^\mathsf{T}\bar{x} + \gamma_i x_i < c^\mathsf{T}\bar{x}$.

Proof. Strictly feasibility (i.e., non-degeneracy) ensures that $\bar{x}_j = ((A^B)^{-1}b)_j > 0$ must hold true. The result then follows trivially. □

When a degenerate pivot is executed, the basis value does not decrease and remains exactly identical. Fortunately Bland's rule (see [2]) avoids the appearance of cycles: the reappearance of an earlier basis.

Proposition 2.7.2. *Let B, N be a given basis, non-basis partition of $\{1, ..., n\}$ with feasible basic solution $\bar{x}$. When employing Bland's rule, no cycles can appear.*

Proof. As a result of Lemma 2.7.2 a cycle, that is a sequence of bases $B_1, ..., B_k, B_1$ must consist of only degenerate pivots. As a result there is a largest index q of a variable entering and leaving the basis during this cycle. Since all variables with larger index still remain as they are, we may as well assume that q is simply the last index altogether. We know that all variables entering and leaving during the cycle have value 0 (they are all degenerate pivots!). Now let us look at the change of basis when q leaves. The new entering variable x_s, with $s < q$ must have negative reduced cost.

We can write the Tableau as follows

$$z = -\bar{z} + \sum_{j \in N} \gamma_j x_j.$$
$$x_k = b_k - \sum_{j \in N} a_{kj} x_j, k \in B.$$

In particular we must have $\gamma_s < 0$, $a_{qs} > 0$ and $b_q = 0$.

However earlier in the cycle when q entered the base, we had $z = -\bar{z} + \sum_{j \in \bar{N}} \bar{\gamma}_j x_j$ and $\bar{\gamma}_q < 0$, but also $\bar{\gamma}_s > 0$ since otherwise s would enter, unless $\bar{a}_{js} \leq 0$ for all $j \in \bar{B}$, but then the problem is shown unbounded and the cycle would have ended. By extending $\bar{\gamma}_j$ with zeros for basic variables, we can also write $z = -\bar{z} + \sum_j \bar{\gamma}_j x_j$.

Consider the matrix $M = [ID]$ and $\bar{M} = [-D^\mathsf{T} I]$, then $\bar{M} M^\mathsf{T} = 0$, in other words every row of $\bar{M}$ is orthogonal to every row of M. The rows of $\bar{M}$ remain orthogonal to rows of any matrix resulting from M when pivoting is applied. In our case,

$$M = \begin{bmatrix} I & 0 & D & b \\ 0 & 1 & \gamma^\mathsf{T} & -\bar{z} \end{bmatrix}.$$

and

$$M^d = \begin{bmatrix} -D^\mathsf{T} & -\gamma & I_r & 0 \\ -b^\mathsf{T} & \bar{z} & 0 & 1 \end{bmatrix}.$$

Applying this principle on the sweeping column and objective row, We arrive at the equation:

$$\gamma_s - \bar{\gamma}_s + \sum_{j \in B} \bar{\gamma}_j a_{js} = 0.$$

But $\gamma_s - \bar{\gamma}_s < 0$ so that $\sum_{j \in B} \bar{\gamma}_j a_{js} > 0$ must hold true. Thus for some $r \in B$, it must be so that $\bar{\gamma}_r a_{rs} > 0$ and in particular $\bar{\gamma}_r \neq 0$, ensuring that $r \in \bar{N}$. However $\bar{\gamma}_q < 0$ and $a_{qs} > 0$, so $r \neq q$. Moreover $a_{js} \leq 0$ for all $j \leq q$ (otherwise Bland's rule would have picked an earlier index) and $\bar{\gamma}_j > 0$ for $j < q$. But we now have arrived at a contradiction. □

Theorem 2.7.2. *Let a linear program in standard form* (2.2) *be given alongside an initially feasible basis B, N. Then the simplex method finitely terminates with either an indication that the problem is unbounded from below or an optimal solution is identified.*

Proof. The set of possible basis' B is finite. Moreover at each pivot operation, the current optimal value (the basis) value decreases strictly or a degenerate pivot occurs. Following Proposition 2.7.2 a sequence of degenerate pivots is finite and thus a subsequent strict decrease of the basis value must occur again. Looking at the subsequence of decreasing pivot operations: should the method not terminate, then an earlier basis must repeat, thus increasing the basis value. This is a contradiction. □

2.7.2 Phase I

Up until now we had assumed available a feasible basis. Now we show how one can set up a version of the simplex method generating such a feasible basis. This is a phase I simplex method. The Phase II simplex method then consists of the method already exposed once a feasible solution has appeared. In the first phase, we simply begin with setting for instance $x = 0$ as an initial solution in (2.1) (and thus $x = 0, s = -b$ in (2.2)).

This leads us to the first Tableau and since it is infeasible, there must exist an index i with $s_i < 0$. Now the infeasible pivot rule is simply:

Definition 2.7.6 (Infeasible pivot)**.** *The infeasible pivot rule consists of picking the smallest i having a currently negative valued basis variable and then the first index j with negative Tableau entry.*

Exercise 2.7.2. *Consider the linear program:*

$$\begin{aligned} \min \quad & -x_1 - x_2 \\ s.t. \quad & -2x_1 - x_2 \leq -4 \\ & x_1 - 2x_2 \leq 1 \\ & 2x_1 + 3x_2 \leq 6 \\ & x \geq 0, \end{aligned}$$

Use the two phase simplex method to solve this problem.

Lemma 2.7.3. *The first phase of the simplex method terminates finitely if the infeasible pivot rule is used.*

Proof. The argument is very similar to our earlier proof of the second phase being finite. Likewise the first phase can only fail to be finite if a cycle occurs. In that case, there is a largest index q entering and leaving the base. We may also assume that q is the largest index altogether. Now when q enters the base and say i leaves the base, we have $b_i < 0$, $a_{ij} > 0$ for all $j < q$ and $a_{jq} < 0$. When q leaves the base, we have $\bar{b}_q < 0$, $\bar{b}_j > 0$ for $j < q$. Now the orthogonality principle dictates that:

$$0 = b_i - \sum_{j=1}^{q} \bar{b}_j a_{ij},$$

but $b_i < 0$, $\bar{b}_j a_{ij} > 0$ for all j, thus entailing that this sum is strictly negative, which is a contradiction. □

Exercise 2.7.3. *Use the simplex method to solve:*

$$\begin{aligned} \min_x \quad & x_1 - 2x_2 \\ s.t. \quad & 2x_1 + 3x_3 = 1 \\ & 3x_1 + 2x_2 - x_3 = 5 \\ & x_1, x_2, x_3 \geq 0. \end{aligned}$$

Exercise 2.7.4. *Use your favourite programming language to implement the simplex method. Download some instances from the Hans Mittelmann's website* `https://plato.asu.edu/ftp/lpopt.html` *and solve to test your code. Use a standard package to read and write MPS files (in python pysmps, in julia the REW file format). N.B. observe that a language with support for linear algebra is preferable. Julia, C++ with Eigen, or F77 are highly recommended ;*

Chapter 3

Integer (Linear) Programming

3.1 The object of study

In practice in many applications it makes sense to assume that certain variables are not continuous. One can think of investment, we can either build something or we can not build it - a half build factory / house is as good as if it was not build at all. Likewise in many situations, we can either switch something on or off, but not half way. In these cases it would make sense to request that certain variables are either 0 or 1. As an extension, certain variables can be naturally assumed to be integer. This leads us to

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & c^\mathsf{T} x \\ \text{s.t.} \quad & Ax \geq b \\ & x \geq 0, x \in \mathbb{R}^{n-p} \times \{0,1\}^p \,, \end{aligned} \tag{3.1}$$

which we will call a "mixed-integer linear program" for simplicity. We could alternatively speak of a "Binary Linear Program" if $p = n$, "Integer Linear Program" if $\{0,1\}$ is replaced with $\mathbb{Z}$ and $p = n$. The word mixed is added if at least one continuous variable is present.

Despite the simple looking problem (3.1), hardly different from (2.1), we have in fact added enormous difficulty. In general problems of class (3.1) are NP-hard, i.e., no known polynomial algorithms exist. This in contrast to (2.1) being a polynomial optimization problem. The version of the simplex method that we have presented is not polynomial, but randomized version of it can be set up. More classically, [3] established a different approach showing polynomiality of the class. The use of interior point methods has since been largely developed [7].

Exercise 3.1.1. *An explorer has a set of items n with given utility $u_1, ..., u_n$ for his exploratory journey. The items have a weight $w_1, ..., w_n$. The explorer has a backpack with capacity C and tries to bring with him the set of items of most utility. Write this problem as a MILP.*

Exercise 3.1.2. *A transportation company must deliver products $P_1, ..., P_m$ of weight $w_1, ..., w_m$ by renting available trucks $T_1, ..., T_n$. Each truck has a weight limit $L_1, ..., L_n$ and can carry at least each item: $\min_{i=1,...,n} L_i \geq \max_{j=1,...,m} w_j$. Renting truck T_i costs r_i. Formulate this problem as a MILP.*

Exercise 3.1.3. *There are n students $S_1, ..., S_n$ that have to choose an advisor from among the k available advisors $A_1, ..., A_k$. Each student i has a preference note $p_{ij} \in [0, 20]$ for advisor j. An advisor can advise no more than 4 students at a time. Express the problem of finding the preference maximizing assignment of students to advisors as a MILP.*

Express the following added conditions (assume $n \geq 4$):

- *Students 1 and 2 want to have the same advisor*
- *Advisor 1 can only advice student 3 if he also advises student 4.*
- *Advisor 2 can only advice an even number of students.*

Definition 3.1.1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuous function, $C \subseteq \mathbb{R}^n$ a closed set. Consider the optimization problem $\min_{x \in C} f(x)$ called (P).*

Assume given functions $\check{f} : \mathbb{R}^n \to \mathbb{R}, \hat{f} : \mathbb{R}^n \to \mathbb{R}$, such that

$$\check{f}(x) \leq f(x) \leq \hat{f}(x), \ \forall x \in \mathbb{R}^n.$$

Assume given moreover closed sets $\hat{C}, \check{C}$ such that $\hat{C} \subseteq C \subseteq \check{C}$.

The optimization problem $\min_{x \in \check{C}} \check{f}(x)$ is called a relaxation of (P). The optimization problem $\min_{x \in \hat{C}} \hat{f}(x)$ is called a restriction of (P).

The simplest case of having relaxations and restrictions keep $\check{f} = f = \hat{f}$, but work with larger (smaller respectively) feasible sets.

Definition 3.1.2. *The optimization problem:*

$$\begin{array}{ll} \min\limits_{x \in \mathbb{R}^n} & c^\mathsf{T} x \\ s.t. & Ax \geq b \\ & x \geq 0, x \in \mathbb{R}^{n-p} \times [0,1]^p, \end{array} \tag{3.2}$$

is called continuous relaxation of (3.1).

Exercise 3.1.4. *Argue that the optimal value of a relaxation is a lower bound on the optimal value of a given optimization problem, whereas the optimal value of the restriction is an upper bound.*

Corollary 3.1.1. *In the setting of Definition 3.1.1, if a given solution for a relaxation is feasible for the original problem, then it is also optimal.*
If for a given $\varepsilon > 0$, and $\hat{x} \in \hat{C}$ feasible for the restriction, $\check{x}$ optimal for the relaxation, we have $\hat{f}(\hat{x}) - \check{f}(\check{x}) \leq \varepsilon$, then $\hat{x}$ is an ε-optimal solution for the problem.

Exercise 3.1.5. *Prove the previous Corollary.*

Let us recall Definition 2.1.6.

Theorem 3.1.1 (Carathéodory)**.** *In the setting of Lemma 2.1.4 we may pick $p = n+1$, i.e., the convex hull of S is given by the convex combination of at most $n+1$ points.*

Proof. Let us assume that for some $x \in \operatorname{Co} S$ we would need $p > n+1$ points. Then with λ becoming a variable of a linear program, the $n \times p$ matrix A populated with columns being the points $x_1, ..., x_p \in S$, $b = x$ we can write for some c

$$\begin{aligned} \min_{\lambda} \quad & c^{\mathsf{T}}\lambda \\ \text{s.t.} \quad & A\lambda = b \\ & e^{\mathsf{T}}\lambda = 1, \\ & \lambda \geq 0. \end{aligned}$$

Now this linear program has a feasible solution by construction and must thus admit an optimal solution being basic: a vertex. But the row-rank can not exceed $n+1$ and so any base has at most $n+1$ non zero variables. □

Exercise 3.1.6. *By leveraging on the example $S = \{(0,0),(0,2),(1,1)\}$ in dimension 2, show that we can not use the convex combination of less than n points, e.g., in dimension 2 the pairswise convex combinations do not suffice.*

3.2 A brief excursion

Definition 3.2.1. *Let $(X, \|\cdot\|)$ be a Banach space, $f : X \to \bar{\mathbb{R}}$ a proper function. Associated with f we define $f^* : X^* \to \bar{\mathbb{R}}$, called Fenchel's conjugate as:*

$$f^*(x^*) = \sup_{x \in X} \{\langle x^*, x\rangle - f(x)\}.$$

Let us now provide some rules of calculus for Fenchel's conjugate and several key identities involving its relationship to f.

Proposition 3.2.1. *Let $(X, \|\cdot\|)$ be a Banach space, $f : X \to \bar{\mathbb{R}}$ a lower semi-continuous proper convex function and f^* Fenchel's conjugate of f. Then it holds:*

1. *The function f^* is convex and (weak*-)lower semi-continuous.*
2. *For any $x \in X, x^* \in X^*$, $f(x) + f^*(x^*) \geq \langle x^*, x\rangle$.*
3. *$f = f^{**}|_X$, and actually if this holds for an arbitrary function f, then f is l.s.c. and convex.*
4. *For any $\alpha \in \mathbb{R}$, $(f + \alpha)^* = f^* - \alpha$.*
5. *For any $\lambda \geq 0$: $(\lambda f)^*(x^*) = \lambda f^*(\frac{x^*}{\lambda})$.*
6. *For a fixed $x_0 \in X$, the shifted map $f_{x_0}(\cdot) = f(\cdot - x_0)$ has Fenchel's conjugate $f^*_{x_0} = f^* + \langle \cdot, x_0\rangle$.*
7. *For any fixed $x_0^* \in X^*$, the shifted map $f_{x_0}(\cdot) = f(\cdot) + \langle x_0^*, \cdot\rangle$ has Fenchel's conjugate $f^*_{x_0}(\cdot) = f^*(\cdot - x_0^*)$.*
8. *For any $x \in X, x^* \in X^*$, $f(x) + f^*(x^*) = \langle x^*, x\rangle$ if and only if $x^* \in \partial f(x)$. Equality also implies that $x \in \partial f^*(x^*)$. When X is reflexive, then $x \in \partial f^*(x^*)$ implies equality holding.*

Proof. Convexity is evident as a suppremum of affine functions. As for (weak* -) lower semi-continuity, let $p^* \in \mathcal{D}\mathrm{om}(f^*)$ be given, together with $\varepsilon > 0$. Then we may find $x \in X$ such that $\langle p^*, x\rangle - f(x) \geq f^*(p^*) - \frac{\varepsilon}{2}$. Moreover by definition of the weak*-topology, we can pick a neighbourhood U of p^* on which $\langle x^* - p^*, x\rangle \leq \frac{1}{2}\varepsilon$ holds true. Combining both shows that for all $x^* \in U$, $\langle x^*, x\rangle - f(x) \geq f^*(p^*) - \varepsilon$, which leads to $f^*(x^*) \geq f^*(p^*) - \varepsilon$ as desired.

In, the third item, as a result of the second, for any $x \in X$, we must have $f(x) \geq f^{**}(x)$. Moreover by the first item, f^{**} is convex and (weakly) lower semi-continuous. Yet Mazur's Theorem yields that f^{**} is actually lower semi-continuous. Now, Hahn-Banach's separation theorem yields the existence of a family $\mathcal{F}$, such that any $\langle x^*, x\rangle - \beta \leq -\alpha$ holds true for all $(x, \beta) \in \mathrm{epi}\, f$, and $(x^*, \alpha) \in \mathcal{F}$. Hence, such $(x^*, -\alpha) \in \mathrm{epi}\, f^*$, for all $(x^*, \alpha) \in \mathcal{F}$. Now for any $(x, \alpha') \in \mathrm{epi}\, f^{**}$, (with $x \in X \subseteq X^{**}$), it holds true that $\alpha' \geq \langle x, x^*\rangle - \alpha$ for all $(x^*, \alpha) \in \mathrm{epi}\, f^*$. Thus in particular, $\alpha' \geq \sup_{(x^*,\alpha)\in\mathcal{F}} \langle x, x^*\rangle + \alpha =$

$f(x)$, by invoking once more Hahn-Banach. Since $(x, \alpha') \in \operatorname{epi} f^{**}$ was arbitrary, it must thus hold that $f^{**}(x) \geq f(x)$ for $x \in X$.

Let us turn attention to the last item. With equality holding, we derive $\langle x^*, y\rangle - f(y) \leq f^*(x^*) = \langle x^*, x\rangle - f(x)$, which by definition means $x^* \in \partial f(x)$. Conversely for any $x^* \in \partial f(x)$, the subgradient inequality yields $f^*(x^*) \leq \langle x^*, x\rangle - f(x)$. But the supremum must be attained at $y = x$.

For any arbitrary $y^* \in X^*$,

$$\begin{aligned} f^*(y^*) &\geq \langle y^*, x\rangle - f(x) = \langle y^* - x^*, x\rangle - f(x) + \langle x^*, x\rangle \\ &\geq \langle y^* - x^*, x\rangle + f^*(x^*), \end{aligned}$$

due to equality holding for the pair (x, x^*) in Fenchel's relationship. Thus by definition $x \in \partial f^*(x^*)$. When X is reflexive, $\partial f^*(x^*) \subseteq X^{**} = X$, so one can argue as before.

The other rules follow simply from the definition and are left as an exercise. □

Corollary 3.2.1. *Let X be a Banach space with dual space X^* and let $C^* \subseteq X^*$ be a given set. Then for a fixed $v \in X$, it holds*

$$\min_{x^* \in C^*} \langle x^*, v\rangle = \min_{x^* \in \operatorname{cl} C^*} \langle x^*, v\rangle = \min_{x^* \in \operatorname{cl}\operatorname{Co} C^*} \langle x^*, v\rangle\,, \tag{3.3}$$

where cl *and* cl Co *refer to the weak* closure and weak*-closed convex hull respectively.*

Proof. The identity trivially holds if C^* is empty, in which case it is already closed and convex. Hence we may assume that C^* is not empty. We have the identity:

$$\min_{x^* \in C^*} \langle x^*, v\rangle = -\sup_{x^* \in X^*} (\langle x^*, -v\rangle - \mathbb{1}_{C^*}(x^*)) = -(\mathbb{1}_{C^*})^*(-v).$$

Now, item 3 of Proposition 3.2.1 indicates that for any function f, (with affine minorant) $f^{**}|_X$ is the l.s.c. convex enveloppe of f, which is equivalent with $\operatorname{cl}\operatorname{Co}\operatorname{epi} f = \operatorname{epi} f^{**}|_X$. By using item 1 and 3 of this same Proposition we establish $f^* = f^{***}$, which thus implies that $f^* = (\operatorname{cl}\operatorname{Co} f)^*$ by what was stated above. With $\operatorname{cl}\operatorname{Co} f \leq \operatorname{cl} f \leq f$ and thus by definition that $(\operatorname{cl}\operatorname{Co} f)^* \geq (\operatorname{cl} f)^* \geq f^*$, it also follows from the last equality that $f^* = (\operatorname{cl} f)^*$. Finally, $\operatorname{epi} \mathbb{1}_{C^*} = C^* \times [0, \infty)$. Hence $\operatorname{cl} \mathbb{1}_{C^*} = \mathbb{1}_{\operatorname{cl} C^*}$ and $\operatorname{cl}\operatorname{Co} \mathbb{1}_{C^*} = \mathbb{1}_{\operatorname{cl}\operatorname{Co} C^*}$ so that the result follows. □

We can use Farkas' Lemma to establish the following result:

Lemma 3.2.1. *Let the polyhedron $P = \{(x_1, x_2) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \ : A_1 x_1 + A_2 x_2 \leq b\}$ be given. Then the projection of P onto the first coordinate, i.e., $P_1 = \{x_1 \ : \ \exists x_2 \text{ s.t. } (x_1, x_2) \in P\}$ is also a polyhedron.*

Proof. For a given $x_1 \in \mathbb{R}^{n_1}$ we have $x_1 \in P_1$ if and only if $A_2x_2 \leq (b - A_1x_1)$ admits a feasible solution. Now Farkas' Lemma 2.4.4 gives us that $x_1 \in P_1$ or there exists $y \geq 0$ such that $A_2^\mathsf{T}y = 0$, $(b - A_1x_1)^\mathsf{T}y < 0$. Let us pick $y_1, ..., y_k$ the vertices of the polyhedron $\left\{y \ : A_2^\mathsf{T}y = 0, y \geq 0\right\}$. Now the claim is that $P_1 = \left\{x_1 \ : y_j^\mathsf{T}(b - A_1x_1) \geq 0, j = 1, ..., k\right\}$. Indeed if some $x_1 \notin P_1$, we can find some y as above, expressible as a convex combination of y_j, and we get $\sum_j \lambda_j y_j^\mathsf{T}(b - A_1x_1) < 0$. □

Corollary 3.2.2. *Let $x_1, ..., x_k$ be a finite collection of points and let $S = \mathrm{Co}\,\{x_1, ..., x_k\}$. Then S is a polyhedron.*

Proof. By making $A = [x_1...x_k]$ and writing $P = \left\{(x, \lambda) \ : [I - A](x, \lambda) = 0, e^\mathsf{T}\lambda = 1, \lambda \geq 0\right\}$. It is clear that P is indeed a polyhedron. Now S is simply the projection of P onto x. □

Corollary 3.2.3 (MILP is LP?)**.** *Consider the MILP* (3.1)*, then it's optimal value ν is also equal to:*

$$\nu = \min_x c^\mathsf{T}x \ s.t. \ x \in \mathrm{Co}\, S,$$

with $S = \{x \in \mathbb{R}^n \ : Ax \leq b, x \geq 0, x \in \mathbb{R}^{n-p} \times \{0, 1\}^p\}$. Moreover $\mathrm{Co}\, S$ is also a polyhedron.

Proof. The only point that needs commenting is why $\mathrm{Co}\, S$ is a polyhedron. If there are only finitely many integer points in S, then S can be expressed as the union of finitely many polyhedra. Then we may employ Corollary 2.2.3 to establish that indeed $\mathrm{Co}\, S$ is itself a polyhedron. Now in the binary case there can only be finitely many integer points and the result is shown. □

Exercise 3.2.1. *Argue that it must follow from this result that computing an explicit representation of a convex hull of a given set is NP-hard in general.*

Admittedly we have not advanced much, but we have gained insights into the structure of MILP.

3.3 Concept of Branch-and-Bound

The idea is now to use a sequence of relaxations in order to solve the MILP. The idea is as follows: we solve a given relaxation of problem (3.1), typically the continuous relaxation. Now if the solution is feasible, we are done. Else there exists an index i, such that $\check{x}_i \notin \{0, 1\}$ (respectively $\check{x}_i \notin \mathbb{Z}$). From the current MILP we will now spawn

two new MILPs, in the first we will impose $x_i = 0$ ($x_i \leq \lfloor \check{x}_i \rfloor$), while in the second we will impose $x_i = 1$ (respectively $x_i \geq \lceil \check{x}_i \rceil$).

The process can be visualized as a tree. Of course we could be wondering if it is needed to branch fully, which would lead to 2^p (in the binary case) leaf-nodes at worst. Fortunately there are options to clean up the tree: pruning.

Lemma 3.3.1 (Dominance)**.** *Let n be a node in the branch and bound tree. Let the optimal value of the problem solved in node n be ν. Any feasible solution to problems in the tree with root node n, $\hat{x}$ must have $c^\mathsf{T}\hat{x} \geq \nu$.*

As a result, if $\nu = \infty$, i.e., the problem is infeasible, then all sub-nodes will have infeasible problems too.

Proof. Since any problem that is spawned from a given other problem is a restriction of the latter, it's optimal value can only increase. This is even more so true when the integer restrictions are restored. □

Corollary 3.3.1 (Pruning / Bounding)**.** *Let n be a node in the branch and bound tree. Let $\bar{x}$ be a given feasible solution to* (3.1)*. Let the optimal value of the problem solved in node n be ν.*

If $\nu > c^\mathsf{T}\bar{x}$, then the subtree with root node n can not contain the optimal solution of problem (3.1)*.*

Proof. This is a direct result of the dominance property. Indeed any feasible solution in the subtree must have $c^\mathsf{T}\hat{x} \geq \nu > c^\mathsf{T}\bar{x}$ and can therefore not be optimal. □

Example 3.3.1. *Let us consider the following linear program:*

$$\begin{aligned} \max_{x,y\in\mathbb{Z}} \quad & 4x + 5y \\ s.t. \quad & x + 4y \leq 10 \\ & 3x - 4y \leq 6 \\ & x, y \geq 0. \end{aligned}$$

The optimal solution of the relaxation is $(4, 1.5)$ with optimal value 23.5. From this father problem we can now form the two children with added constraints $y \leq 1$, $y \geq 2$ respectively. It is now immediately clear that the right problem leads to the newly optimal solution $(2, 2)$ with optimal value 18, whereas the left one leads to $(3\frac{1}{3}, 1)$ with optimal value $18\frac{1}{3}$. It is not needed to explore further the right node, but the left node may still exhibit a better solution. Hence we branch again on $x \leq 3$ and $x \geq 4$ this time.

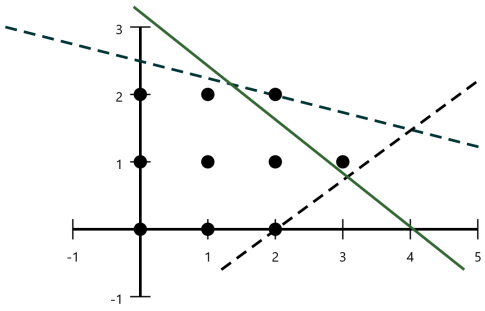


Figure 3.1: Feasible set of the example

It is clear that the subproblem with added constraints $x \geq 4, y \leq 1$ *is infeasible. However the problem with added constraints* $x \leq 3, y \leq 1$ *leads to optimal solution* $(3,1)$ *with optimal value* 17. *We have thus identified the optimal solution* $(2,2)$ *with optimal value* 18. *Observe moreover that we needed an extra layer of branching before having proven that this is indeed so!*

Exercise 3.3.1. *Sketch the Branch and bound tree of the previous example.*

Theorem 3.3.1. *Assume given bounds on the integer variables. Then, the Branch and Bound procedure terminates finitely.*

Proof. Since the Branch and bound procedure can at worst lead to the full exploration of the combination of all integer variables, at worst that many LPs have to be solved. □

It is not true however that in all cases the B&B procedure terminates:

Example 3.3.2. *Consider the problem:*

$$\begin{array}{ll} \min\limits_{y \in \mathbb{Z}^2} & 0 \\ s.t. & 1 \leq 3y_1 - 3y_2 \leq 2, \end{array}$$

which is integer infeasible. However in the branch and bound tree with added constraints $y_1 \geq k, y_2 \geq \ell$*, with* $l \in \{k-1, k\}$*, the points:* $(k, \frac{3k-2}{3})$ *is feasible when* $\ell = k-1$ *and* $(\frac{3k+2}{3}, l)$ *is feasible when* $\ell = k$.

Up until now we have seen cases where only one variable applied for branching. Evidently in larger problems one may have many variables applying for branching. A specific strategy for branching is called a branching strategy. Some options include:

- picking the variable that is closest to being halfway in between two integers.

- the most impactfull variable (e.g., binary, or from a modelling perspective)
- the variable with the most impact in the objective function

Moreover when Branch and Bound is performed on a sequential machine, the to-be solved problems stack up in a queue of problems to be solved. A strategy has to be conceived to handle these pending problems. This could for instance be done, by a depth first, breadth first or best-first strategy. The last amounts to choosing to add new problems, based on the value of the relaxation. In other words, all relaxations are solved, and only the one with the best lower bound is branched upon. The underlying idea is that of a queue of problems to solve: those that have been generated by "branching". The pending problem question is thus implicitly one of how problems are inserted into the queue or which one is first popped from it.

Exercise 3.3.2. *Solve the following MILP with the Branch and Bound algorithm:*

$$\begin{aligned} \min_{x} \quad & x_1 - x_2 \\ s.t. \quad & -2x_1 + 2x_2 \geq 1 \\ & -8x_1 + 10x_2 \leq 13 \\ & x_1, x_2 \geq 0, x_1, x_2 \in \mathbb{Z} \end{aligned}$$

3.4 Strategies with cuts

As observed in Corollary 3.2.3, the MILP (3.1) can be written in the form of a LP. This provides a fundamental further device for solving. Indeed, as is known (Hahn-Banach), any $\bar{x} \in \mathbb{R}^n$ and any closed convex set $S \subseteq \mathbb{R}^n$ with $\bar{x} \notin S$ can be separated. There exists thus a half-space H such that $x \notin H$ and $S \subseteq H$. The inequality $a^\mathsf{T} x \leq b$ defining the half-space is called a valid inequality and in fact a "cut" since it ensures that $\bar{x}$ is removed from the feasible set.

The abstract device producing a valid inequality that cuts off a given fixed point is called a separation mechanism.

Definition 3.4.1. *A given inequality $a^\mathsf{T} x \leq b$ is called supporting for a closed convex set S if any $x \in S$ satisfies the inequality and at least one $\bar{x} \in S$ saturates the inequality.*

For specifically structured problems it is possible to identify valid inequalities.

3.4.1 Gomory Cuts

Theorem 3.4.1 (Gomory cuts)**.** *Let the MILP* (3.1) *be given. Let us be given a base variable x_r that is not integer, that with the help of the simplex Tableau satisfies:*

$$\tilde{b}_r = x_r + \sum_{j \in N} \tilde{a}_{rj} x_j,$$

with N the non-basic variables. Let $J_C \cup J_E$ be the partition of the set of indexes in continuous and integer restricted variables respectively. Let us set $f_0 = \tilde{b}_r - \left\lfloor \tilde{b}_r \right\rfloor$ and $f_j = \tilde{a}_{rj} - \lfloor \tilde{a}_{rj} \rfloor$ and define

$$\begin{aligned} J_E^+ &= N \cap J_E \cap \{j \ : f_j > f_0\} \\ J_E^- &= N \cap J_E \cap \{j \ : f_j \leq f_0\} \\ J_C^+ &= N \cap J_C \cap \{j \ : \tilde{a}_{rj} \geq 0\} \\ J_C^- &= N \cap J_C \cap \{j \ : \tilde{a}_{rj} < 0\}. \end{aligned}$$

Then

$$\sum_{j \in J_E^-} f_j x_j + \sum_{j \in J_E^+} \frac{f_0(1-f_j)}{1-f_0} x_j + \sum_{j \in J_C^+} \tilde{a}_{rj} x_j - \sum_{j \in J_C^-} \frac{f_0}{1-f_0} \tilde{a}_{rj} x_j \geq f_0$$

is a valid inequality.

Proof. From the simplex tableau equation, upon filtering out J_C^+, we can arrive at the valid inequality:

$$x_r + \sum_{j \in J_E^+ \cup J_E^-} \tilde{a}_{rj} x_j + \sum_{j \in J_C^-} \tilde{a}_{rj} x_j \leq \tilde{b}_r.$$

This can be written in short hand as:

$$u + \alpha_1 v_1 + \alpha_2 v_2 \leq \beta + w, \tag{3.4}$$

with $\alpha_1 v_1 = \sum_{j \in J_E^+} \tilde{a}_{rj} x_j$, $\alpha_2 v_2 = \sum_{j \in J_E^-} \tilde{a}_{rj} x_j$, $w = -\sum_{j \in J_C^-} \tilde{a}_{rj} x_j$, $\beta = \bar{b}_r$ and $u = x_r$. The inequality

$$u + \lfloor \alpha_1 \rfloor v_1 + (\lfloor \alpha_2 \rfloor + \frac{\{\alpha_2\} - \{\beta\}}{1 - \{\beta\}} v_2 \leq \lfloor \beta \rfloor + \frac{w}{1 - \{\beta\}}, \tag{3.5}$$

is valid for (3.4) provided $\{\alpha_1\} \leq \{\beta\} \leq \{\alpha_2\}$ with $\{\alpha_1\} = \alpha_1 - \lfloor \alpha_1 \rfloor$.

First we observe that $\lceil \alpha_2 \rceil = -(1 - \{\alpha_2\}) = \alpha_2$ and likewise for the other terms. Moreover $\{\alpha_1\} v_1 \geq 0$, so that we obtain from (3.4):

$$u + \lfloor \alpha_1 \rfloor v_1 + \lceil \alpha_2 \rceil v_2 \leq \beta + w + (1 - \{\alpha_2\}) v_2, \tag{3.6}$$

and the left-hand side is integer valued. We now obtain

$$u + \lfloor \alpha_1 \rfloor v_1 + \lceil \alpha_2 \rceil v_2 \leq \lceil \beta \rceil - (1 - \{\beta\}) + w + (1 - \{\alpha_2\})v_2,$$

and divide by $(1 - \{\beta\}) > 0$ to obtain:

$$\frac{1}{1 - \{\alpha_2\}}(u + \lfloor \alpha_1 \rfloor v_1 + \lceil \alpha_2 \rceil v_2 - \lceil \beta \rceil) \leq -1 + \frac{w + (1 - \{\alpha_2\})v_2}{1 - \{\beta\}}.$$

Now $u + \lfloor \alpha_1 \rfloor v_1 + \lceil \alpha_2 \rceil v_2$ is integer and either we have $u + \lfloor \alpha_1 \rfloor v_1 + \lceil \alpha_2 \rceil v_2 \leq \lfloor \beta \rfloor$ or the term is greater than $\lceil \beta \rceil$. In the second case we have the estimate: $u + \lfloor \alpha_1 \rfloor v_1 + \lceil \alpha_2 \rceil v_2 - \lceil \beta \rceil \leq \frac{1}{1-\{\alpha_2\}}(u + \lfloor \alpha_1 \rfloor v_1 + \lceil \alpha_2 \rceil v_2 - \lceil \beta \rceil)$, leading to the given inequality. In the other case, one can immediately conclude from (3.6) by taking the lower integer of β in the rhs therein.

We can now apply this inequality to derive

$$x_r + \sum_{j \in J_E^+} \lfloor \tilde{a}_{rj} \rfloor x_j + \sum_{j \in J_E^-} \left(\lfloor \tilde{a}_{rj} \rfloor + \frac{f_j - f_0}{1 - f_0} \right) x_j + \sum_{j \in J_C^-} \frac{\tilde{a}_{rj}}{1 - f_0} x_j \leq \left\lfloor \tilde{b}_r \right\rfloor.$$

This inequality can be substracted from the simplex tableau equality:

$$x_r + \sum_{j \in J_E^+} \lfloor \tilde{a}_{rj} \rfloor x_j + \sum_{j \in J_E^-} \lfloor \tilde{a}_{rj} \rfloor x_j + \sum_{j \in J_C^+} \lfloor \tilde{a}_{rj} \rfloor x_j + \sum_{j \in J_C^-} \lfloor \tilde{a}_{rj} \rfloor x_j = \tilde{b}_r,$$

to arrive at the desired case. □

Exercise 3.4.1. *Consider the problem:*

$$\begin{aligned} \min_x \quad & -10x_1 - 5x_2 \\ s.t. \; & 8x_1 + 5x_2 \leq 43 \\ & 5x_1 + 12x_2 \leq 51 \\ & x_1, x_2 \in \mathbb{N}. \end{aligned}$$

Solve the continuous relaxation and compute the Gomory cut corresponding to that solution (i.e., that base).

3.4.2 Other cuts

Lemma 3.4.1 (No good cuts)**.** *Assume given the problem* (3.1) *and that all variables have a binary restriction. Then for a given unfeasible $\bar{x}$, the inequality:*

$$\sum_{i \,:\bar{x}_i = 0} x_i + \sum_{i \,:\bar{x}_i = 1} (1 - x_i) \geq 1,$$

is a valid cut.

Exercise 3.4.2. *Establish formally the previous result.*

Proposition 3.4.1 (Feasibility cuts)**.** *Consider the MILP written as:*

$$\begin{aligned} \min_{x,y} \quad & c^\mathsf{T} x + q^\mathsf{T} y \\ s.t. \quad & Tx + Wy \leq h \\ & x \in \mathbb{Z}^n, y \in \mathbb{R}^p_+, \end{aligned}$$

Then $V : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ defined as $V(x) = \min_{y \geq 0} q^\mathsf{T} y$ s.t. $Wy \leq h - Tx$ is an extended valued convex polyhedral function. Then at a given $\bar{x}$, either this problem is feasible or there exists $\lambda \geq 0$, such that $W^\mathsf{T}\lambda \geq 0$ and $-(h - T\bar{x})^\mathsf{T}\lambda > 0$.

Moreover:

$$\mathcal{D}\mathrm{om}(V) \subseteq \left\{ x \in \mathbb{R}^n \ : (h - Tx)^\mathsf{T}\lambda \geq 0 \right\}, \tag{3.7}$$

where λ is as above (recall $V(\bar{x}) = \infty$).

Proof. The exclusivity condition is just an application of Farkas Lemma 2.4.4 with $A = -W$, $b = -(h - Tx)$. Now for the second item, let us assume given $\bar{x}$ with $V(\bar{x}) = \infty$ and hence λ as indicated. Let us assume moreover that $x \in \mathcal{D}\mathrm{om}(V)$ is given but that $(h - Tx)^\mathsf{T}\lambda < 0$, i.e., $-(h - Tx)^\mathsf{T}\lambda > 0$. Let $\bar{\lambda}$ be some arbitrary dual feasible solution, i.e., $W^\mathsf{T}\lambda \geq -q$, which must exist, as a result of Theorem 2.4.1. Indeed recall that the primal problem is feasible by assumption $x \in \mathcal{D}\mathrm{om}(V)$. In particular $\bar{\lambda} + \alpha\lambda$ is dual feasible for all $\alpha > 0$. Weak duality now yields:

$$-(h - Tx)^\mathsf{T}\bar{\lambda} + \alpha(-(h - Tx)^\mathsf{T}\lambda) \leq q^\mathsf{T} x < \infty,$$

which is evidently a contradiction since $-(h - Tx)^\mathsf{T}\lambda > 0$ and α can grow indefinitely. So, $(h - Tx)^\mathsf{T}\lambda \geq 0$ must hold true and the result is shown. □

Remark 3.4.1. *In the setting of Proposition 3.4.1 we can now consider solving the MILP as $\min_x c^\mathsf{T} x + V(x)$, wherein typically V would be replaced by a cutting plane model for it. This is the concept of Benders decomposition [1]: a variant of Kelley's cutting plane method [4]. For more details and exploitation of special structure we refer to [10].*

3.5 Special structure

The question we will now address is wether or not it is possible for the continuous relaxation to immediately yield an integer solution. The presence of a special matrix structure will provide some insights in this case.

Definition 3.5.1. *A $m \times n$ matrix A is said to be unimodular if all of it's square $m \times m$ submatrices B we have* $\det B \in \{-1, 0, 1\}$.

The matrix is moreover said to be totally unimodular if the same holds true for all $p \times p$ submatrices.

Proposition 3.5.1. *Let A be a $m \times n$ matrix with integer entries and rank m. Then the following statements are equivalent:*

- *A is unimodular*
- *The vertices of the polyhedron $P = \{x \in \mathbb{R}^n \ : Ax = b, x \geq 0\}$ are integer for all integer vectors b.*
- *Each $m \times m$ square non-singular sub-matrix B of A has an inverse matrix B^{-1} with integer entries.*

Proof. Point 3 clearly implies 2 since any vertex x of P results from the identification of a base, in other words, non-singular submatrix B, such that $x = B^{-1}b$. The inverse implication follows upon realizing that each vertex of P results from the identification of some non-singular submatrix B of A, such that $x = B^{-1}b$. Now since this must be so for all integer vectors b, in particular the unit-vectors, B^{-1} itself can only contain integer elements.

The equivalence of point 1 and 2 follows from the use of Cramer's rule. Indeed for $x \in P$ to be a vertex, amounts to x being the unique solution of $A_I x = b_I$ for some selection I of rows. Now this A_I can be reduced back to (the non-basic solutions are set to zero) a square submatrix and can not have zero determinant, since then the solution would not be unique ; Cramer's rule now gives us that x_i the solution of this equation is equal to $\frac{\det(A_i)}{\det A}$ with A_i the matrix resulting from A by replacing the ith column with b. It is now clear that x_i is integer. The converse is now also clear. □

Proposition 3.5.2. *Let A be a $m \times n$ matrix with integer entries and rank m. Then the vertices of $P = \{x \in \mathbb{R}^n \ : Ax \leq b, x \geq 0\}$ are integer for all integer vectors b if and only if A is totally unimodular.*

Proof. Result originally due to [6] □

Proposition 3.5.3. *Let A be a $m \times n$ matrix with entries taking values in $\{0, 1, -1\}$. Then A is totally unimodular if*

- *Every column contains at most two non-zero coefficients.*

- *the rows of A can be partitioned into two sets R_1, R_2:*
  - *if a column j contains two elements $a_{ij} \neq 0$, $a_{hj} \neq 0$ with the same sign, then $i \in R_1, h \in R_2$.*
  - *if a column j contains two elements $a_{ij} \neq 0$, $a_{hj} \neq 0$ with the different sign, then both i, h belong to the same index set.*

3.6 Products of variables, reformulations

Let us now turn our attention to a very powerful use of binary variables: notably in representing disjunctive or logical implications. Likewise binary variables can be employed to represent piecewise linear mappings. It is also possible to replace products of binary variables, as well as products with binary variables by alternative linear representations.

Lemma 3.6.1. *Consider an optimization problem featuring the product of binary variables: $\prod_{i\in I} x_i$, $x_i \in \{0,1\}, i \in I$. Then this product can be replaced by an additional binary variable w and the additional constraints:*

$$\begin{aligned} & w \leq x_i, i \in I \\ & w \geq \sum_{i\in I} x_i - |I| + 1. \end{aligned}$$

Proof. One readily verifies that $w = 0$ if and only if there exists $i \in I$ with $x_i = 0$ as well as $w = 1$ if and only if all $x_i = 1$. □

Exercise 3.6.1. *Argue that the optimization problem:*

$$\begin{aligned} \min_x & \frac{1}{2} x^\mathsf{T} Q x + c^\mathsf{T} x \\ s.t. & Ax \leq b, \\ & x \in \{0,1\}^n, \end{aligned}$$

can be written as a BLP, regardless of the properties of the $n \times n$ matrix Q. Write down the new representation.

Exercise 3.6.2. *A company receives m orders $O_1, ..., O_m$ and has manufactured products $P_1, ..., P_n$ with associated weights $w_1, ..., w_n$. The orders consists of a request $d_1, ..., d_m$ of a certain amount of weight. The company wishes to honour each order while shipping a minimum total weight. Formulate the problem as a MILP.*

Now assume moreover that if product P_1 is attributed to order O_1, that product P_2 and P_3 must also be attributed to order O_1. Formulate this added constraint and ensure that the resulting problem remains a MILP.

Lemma 3.6.2. *Consider an optimization problem featuring the product of binary variable x and continuous bounded variables y, $y \in [\underline{y}, \overline{y}]$. Then the product xy can be replaced by an additional continuous variable z and the following constraints*

$$\begin{aligned} \underline{y}x \leq z \leq x\overline{y} \\ y + \overline{y}(x-1) \leq z \leq y + \underline{y}(x-1) \\ z \in \mathrm{Co}\left(\{0\} \cup [\underline{y}, \overline{y}]\right). \end{aligned}$$

Proof. Much like the earlier case, the substitution is readily evaluated to be exact. □

An important feature in modelling is sometimes the necessity to be able to enforce one or another constraint. This amounts to have a constraint that we would like to hold whenever a given binary variable is true. In fact, implicitly the binary variable is defined by the constraint holding:

$$u = 1 \iff a^{\mathsf{T}} x \leq b.$$

Now this too can be modelled using the "big-M" idea. To this end let us be given

$$M = \sup_{x \in X} a^{\mathsf{T}} x - b,$$

with X for instance bounds on the variables X (or ideally more restrictions). Then we can simply write:

$$a^{\mathsf{T}} x \leq b + M(1-u)$$

as a valid inequality for all $x \in X$ and the logical implication given earlier. The constant M is typically referred to as a "big-M". It turns out that usually when M is "large", the formulations are weak.

An interesting use of the idea is when considering so called selection problems. For instance,

$$\begin{aligned} \min_{x, I \subseteq \{1,\ldots,m\}} \quad & c^{\mathsf{T}} x \\ \text{s.t.} \quad & a_i^{\mathsf{T}} x \leq b, i \in I, \\ & |I| \geq p. \end{aligned}$$

In the previous problem one needs to select p among m possible constraints to enforce, all while minimizing $c^\mathsf{T}x$. The latter type of problem arises as a popular approach for discrete versions of so-called chance-constrained problems (see [5,8]). The discretisation brings about other negative side effects, such as the obtained solution frequently being infeasible for the undiscretized object.

Binary variables are also very useful in expressing logical conditions. For instance

- $A \vee B$ can be expressed as $x_1 + x_2 \geq 1$.
- $A \wedge B$ can be expressed as $x_1 = x_2 = 1$.
- $\neg A$ can be expressed as $x_1 = 0$
- A implies B as $x_1 \leq x_2$,
- A if and only if B as $x_1 = x_2$.

A further interesting use of binary variables is the representation of a piecewise linear or affine map.

Definition 3.6.1. *A mapping $f : \mathbb{R}^n \to \mathbb{R}$ is called piecewise affine (linear) if a finite collection $\mathcal{I}$ of intervals can be identified, that cover $\mathbb{R}^n$ and such that for $I \in \mathcal{I}$, we can identify $a_I \in \mathbb{R}^n$, $b_I \in \mathbb{R}$, such that $f(x) = a_I^\mathsf{T}x - b_I$ for all $x \in I$.*

A piecewise linear map can thus be represented, with $m = |\mathcal{I}|$ binary variables as follows. First let $\mathcal{I} = \{I_1, ..., I_m\}$ with $I_j = [\underline{x}_j, \bar{x}_j]$, where the interval is understood as potentially degenerate and componentswise. We now write

$$\begin{aligned} \underline{x}_j u_j \leq y_j \leq \bar{x}_j u_j, j = 1, ..., m \\ \sum_{j=1}^{m} u_j = 1 \\ u_j \in \{0,1\}, j = 1, ..., m \\ a_j^\mathsf{T} y_j - u_j b_j = v_j, j = 1, ..., m \\ \sum_{j=1}^{m} y_j = x \end{aligned}$$

and $f(x)$ can be replaced by $\sum_{j=1}^m v_j$, in fact the latter variables can be substituted out.

It is also possible to reformulate absolute values with additional variables. For instance the function $f(x) = |x|$ can be replaced with the additional variable v, $x \leq v, -x \leq v$. This reformulation does not require binary variables at all.

Exercise 3.6.3. *We are considering a project that has two independent activities A_1 and A_2. The duration d_1 of the first activity is somewhere in between 2 and 4 hours. The duration of the second activity d_2 varies between 1 and 4 hours. The cost of the first activity is $11-2.5d_1$, the cost of the second $9-2d_2$. The project is completed when both activities have been executed. An activity can not begin when another has not been completed.*

Write the resulting problem of finding the optimal task duration d_1, d_2 and starting times s_1, s_2 as a combinatorial program with logical restrictions.

Next write the problem as a MILP.

Exercise 3.6.4. *Use your favourite programming language to implement a Branch & Bound and/or Branch& Cut approach; Download some easy instances from the miplib (`https://plato.asu.edu/ftp/milp.html`) and solve to test your code. Again a langage like Julia is recommended (especially over python).*

Chapter 4

A primer in Graphs

4.1 Basic concepts

Closely related to discrete or combinatorial optimization is the concept of graph and graph theory. We will only be able to briefly sketch some concepts.

Definition 4.1.1. *A graph $G = (V, A)$ is a collection of nodes V and arcs A. The set of arcs A can be understood as pairs of nodes: $A \subseteq V \times V$. Arcs can be directed or undirected.*

Definition 4.1.2. *Let a graph $G = (V, A)$ be given. The graph $G' = (V', A')$ is called a subgraph of G if $V' \subseteq V$ and $A' \subseteq V' \times V'$.*

A subset V' of V is said to generate subgraph (V', A') if A' consists of the selection of all arcs A having both endpoints in V'.

Definition 4.1.3. *Let a graph $G = (V, A)$ be given. A subset V' of V is said to be an independent set if the subgraph generated by it has empty arc set. The independence number of a graph is the cardinality of the largest possible independent set.*

Definition 4.1.4. *Let a graph $G = (V, A)$ be given. The graph is called complete if A connects any two pair of vertices. A clique is a node subset V' of V generating a complete subgraph.*

Definition 4.1.5. *Let a graph $G = (V, A)$ with directed arcs be given. For a given node $n \in V$, the set of ancestor nodes $\mathcal{A}(n) = \{v \in V : (v, n) \in A\}$. The set of children nodes: $\mathcal{F}(n) = \{v \in V : (n, v) \in A\}$.*

Exercise 4.1.1. *Argue that the consideration of a directed graph does indeed make a difference in the previous definition. Indeed in an undirected graph we would have $\mathcal{A}(n) = \mathcal{F}(n)$.*

Definition 4.1.6. *Let a graph $G = (V, A)$ be given. A partition V_1, V_2 of V is called a bipartition if A only contains arcs not having both endpoints in V_1 or V_2, i.e., one end point must be in V_1, the other in V_2.*

Example 4.1.1. *The classic example of a bipartition is the marriage problem, wherein V_1 would be the collection of men, V_2 the collection of women and arcs attach a unique node in V_2 to one in V_1.*

Example 4.1.2. *Another example of a problem involving a bipartition is the so-called bichromatic coloring problem. Can one assign one of two colours to each node, in such a way that no connected nodes have the same colour. This can be done if and only if the graph admits a bipartition.*

Definition 4.1.7. *Let a graph $G = (V, A)$ with directed arcs be given. A $|V| \times |A|$ matrix M is called incidence matrix if for a given column $e = (i, j)$, $M_{ie} = -1, M_{je} = 1$.*

Exercise 4.1.2. *Let a graph $G = (V, A)$ be given. The colouring problem consists of attributing to each node a unique colour such that no adjacent nodes have the same colour. The chromatic number of a graph is the minimum number of such colours needed. Write this problem as a MILP.*

4.2 Flows

Definition 4.2.1. *Let a directed graph $G = (V, A)$ be given. Let us be given equally a capacity vector $c \geq 0$, $c \in \mathbb{R}^{|A|}$, demand vector $d \in \mathbb{R}^{|V|}$. Then a (feasible) flow vector of (G, c, d) is a vector $x \in \mathbb{R}^{|A|}$ such that*

$$\begin{aligned} 0 \leq x_a \leq c_a, \ \forall a \in A \\ d_n = \sum_{v \in \mathcal{A}(n)} x_{(v,n)} - \sum_{v \in \mathcal{F}(n)} x_{(n,v)}, \ \forall n \in V. \end{aligned}$$

Definition 4.2.2. *Let a directed graph $G = (V, A)$ be given alongside a capacity vector c, demand d and cost vector w. Then the problem of minimizing $w^\mathsf{T} x$ over all feasible flows (G, c, d) is called a min cost-flow problem.*

Lemma 4.2.1. *Let M be the incidence matrix of the directed graph $G = (V, A)$. Then the min cost-flow problem can be written as:*

$$\begin{aligned} \min_{x \in \mathbb{R}^{|A|}} \quad & w^\mathsf{T} x \\ s.t. \quad & Mx = d \\ & 0 \leq x \leq c. \end{aligned}$$

If moreover the flow vector has to be integer, then the previous problem is combinatorial. It is pretty clear that the incidence matrix is totally unimodular by definition. The same holds true for the extended matrix

$$\begin{bmatrix} M \\ -M \\ I \end{bmatrix}$$

As a result Proposition 3.5.2 allows us to solve the continuous relaxation (provided d, c are integer) to obtain immediately the optimal integer feasible flow.

Exercise 4.2.1. *Let a directed graph $G = (V, A)$ be given with capacity vector c and a set of commodities K. Each commodity $k \in K$ corresponds to a triplet (s_k, t_k, d_k) with s_k the source node of the commodity, t_k the destination node of the commodity and d_k the demand for the commodity. For each arc $a \in A$, and commodity $k \in K$, $w_a^k \geq 0$ is the cost of sending commodity k over arc a. Write the optimization problem that minimizes total costs of shipping the commodities, while ensuring that capacity of the arcs is not exceeded.*

4.3 Shortest paths

Definition 4.3.1. *Let a directed graph $G = (V, A)$ be given alongside an arc-cost vector w. Let two nodes $s, t \in V$ be set aside. A path from s to t is a collection of arcs $a_1, ..., a_k$ with $a_1 = (s, v)$, $a_2 = (v, .)$, ..., $a_k = (n, t)$, i.e., where each successive arc couple $a_i, a_{i+1} = (v, n), (n, w)$ is such that $v \in \mathcal{A}(n), w \in \mathcal{F}(n)$.*
Then the problem of finding the cost minimal path is:

$$\begin{aligned} \min_{P} \quad & \sum_{a \in P} w_a \\ s.t. \quad & P \text{ is a path from } s \text{ to } t \end{aligned}$$

Definition 4.3.2. *A path is called simple if it does not contain the same arc more than once. A path is elementary if no node is visited twice. A path is said to contain a cycle if we can extract from it a nonempty subpath emanating and terminating in the same node.*

Definition 4.3.3. *A graph $G = (V, A)$ is said to be connected if any two vertices in V are connected through a path. A connected component of G is the largest connected subgraph.*

Definition 4.3.4. *A tree is a connected graph having no cycles and undirected arcs. A spanning tree of a given graph $G = (V, A)$ is a tree with vertex set V and subselection of arcs A. A forest is a collection of disjoint trees.*

Remark 4.3.1. *A tree is a set of nodes and arcs such that there is exactly a unique path between any two nodes in the tree. For a forest, there is at most one path between any two nodes. Any subselection of nodes and arcs of a forest (even if it is a single tree) is therefore also a forest.*

Exercise 4.3.1. *Let a connected graph $G = (V, A)$ be given. Establish that it must admit a spanning tree.*

Exercise 4.3.2. *Let M be the incidence matrix of a graph $G = (V, A)$ and let M_T be the set of columns of M associated with the set of arcs $T = (a_1, ..., a_q)$ of A. Establish that M_T has all columns linearly independent iff T defines a forest of G.*

Lemma 4.3.1. *The rank of an incidence matrix of a connected directed graph is $|V|-1$.*

Proof. The elements in each row sum to zero. Hence the null space of the rows is at least of dimension 1. It follows that the rank of M is not larger than $|V|-1$. Since the graph is connected there exists a spanning tree T. Consequently due to the previous exercise the columns M_T are linearly independent and there are $|V|-1$ of these. It follows that the rank of M is not smaller than $|V|-1$. □

Lemma 4.3.2. *Let a directed graph $G = (V, A)$ be given alongside an arc-cost vector w. Consider the binary vector $x \in \{0,1\}^{|A|}$. Then the constraints:*

$$\begin{aligned}
\sum_{v \in \mathcal{A}(s)} x_{(v,s)} &= 0 \\
\sum_{v \in \mathcal{F}(s)} x_{(s,v)} &= 1 \\
\sum_{v \in \mathcal{A}(t)} x_{(v,s)} &= 1 \\
\sum_{v \in \mathcal{F}(t)} x_{(s,v)} &= 0 \\
\sum_{v \in \mathcal{A}(n)} x_{(v,n)} &= \sum_{v \in \mathcal{F}(n)} x_{(n,v)},\ \forall n \in V \setminus \{s,t\}
\end{aligned}$$

define a valid path.

Corollary 4.3.1. *If the vector d is defined as $d_s = -1$, $d_t = 1$, $d_u = 0$ elsewhere. Then the minimum path problem is also a min-cost flow problem of the type* $\min_x w^\mathsf{T} x$ *s.t.* $Mx = d, x \geq 0$.

The dynamic programming algorithm is a key method for solving shortest path problems. To this end let us attribute a value function $\nu : V \to \mathbb{R} \cup \{\infty\}$, wherein for $v \in V$, $\nu(v)$ is the length of the shortest path from v to t.

Proposition 4.3.1 (Bellman's optimality principle)**.** *Let a directed graph $G = (V, A)$ be given alongside an arc-cost vector w. Assume set aside nodes $s, t \in V$ and assume given a cost-minimal path p from s to t. Should p pass through a node $n \in V$, not being infinitely long before, then q, the subpath of p from n to t must be cost-minimal from n to t.*

Proof. Let us assume by contradiction that this is not so. The path $p = (a_1, ..., a_k, a_{k+1}, ..., a_r)$ with $a_j \in A$ arcs and a_k having endpoint n, a_{k+1} having n as beginning. If the cost-optimal path is not finite, it must contain a cycle, and effectively the problem is unbounded from below, because the cycle contributes an effective negative value: $\sum_{a \in c} w_a < 0$ for the cycle c (otherwise the optimal solution would not take the cycle). When the cycle sums to zero, an infinite repetition of it can be simply deleted, essentially making p a finite path.

Now $q = (a_{k+1}, ..., a_r)$ and should this not be the cost-optimal path, i.e., should we be able to find a path q' from n to t with strictly smaller length, then since $(a_1, ..., a_k, q')$ is now a path from s to t with strictly smaller length, unless the path $(a_1, ..., a_k)$ is of cost $-\infty$, but we have excluded it to be infinite. We have thus reached a contradiction with p being cost-minimal. □

Proposition 4.3.2. *Let a directed graph $G = (V, A)$ be given alongside an arc-cost vector w. The value function ν satisfies the following recursive relationship:*

$$\begin{aligned} \nu(t) &= 0 \\ \nu(v) &= \min_{a=(v,n)\in A} \{w_a + \nu(n)\}, v \neq t \end{aligned}$$

Proof. The initialisation is clear by definition. Now let us be given a cost-minimal path p from node v to t so that $\nu(v) = \sum_{a \in p} w_a$. There exists at least one $n \in \mathcal{F}(v)$ such that (v, n) is the first arc of the path p. We can thus write $p = ((v, n), p')$ and $\nu(v) = w_{(v,n)} + \sum_{a \in p'} w_a$. But now by Proposition 4.3.1 p' must be the cost-minimal path from node n to t. Hence $\nu(v) = w_{(v,n)} + \nu(n) = \min_{a=(v,n)\in A} w_a + \nu(n)$. □

Definition 4.3.5. *The previously defined value function is frequently called a Bellman function.*

Exercise 4.3.3. *Think of a simple computational procedure evaluating the value function to find the shortest path.*

Definition 4.3.6. *Let a graph $G = (V, A)$ be given alongside a weight vector $w \in \mathbb{R}^{|A|}$. The problem of finding the minimum cost spanning tree consists of identifying $T = (V, A')$ such that $\sum_{a \in A'} w_a$ is minimal.*

We have Kruskal's Algorithm to solve this problem:

- (Initialisation): Sort the weight vector by increasing weight. The initial set of arcs $A' = \emptyset$, set $i = 1$.
- (Stopping test) If $|A'| \geq |V| - 1$ stop, else continue
- (Check): If $(V, A' \cup \{a_i\})$ has no cycle, then add a_i to A'.
- (Update): Set $i = i + 1$ and return to the stopping test.

Exercise 4.3.4. *Consider the complete graph $G = (V, A)$ with $V = \{1, ..., 5\}$ and weight vector $w_{(1,2)} = 1, w_{(1,3)} = 9, w_{(1,4)} = 5, w_{(1,5)} = 7$, $w_{(2,3)} = 6, w_{(2,4)} = 10, w_{(2,5)} = 3$, $w_{(3,4)} = 2, w_{(3,5)} = 2$, $w_{(4,5)} = 2$. Apply Kruskal's Algorithm to identify a spanning tree.*

Chapter 5

Project

5.1 General description

The Unit Commitment problem in energy management aims at finding the optimal production schedule of a set of generation units, while meeting various system-wide constraints. It has always been a large-scale, non-convex, difficult problem, especially in view of the fact that, due to operational requirements, it has to be solved in an unreasonably small time for its size. In this project we will play with this problem a bit. We refer the reader to [9] and references therein for detailed information.

From an abstract perspective we are given a set of assets $i = 1, ..., m$ capable of generating power. These assets consist of thermal plants, cascading reservoir systems, renewable intermittent generators. Each asset comes with an abstract feasible set $X_i \subseteq \mathbb{R}^{n_i}$ assumed closed, cost function $c_i : \mathbb{R}^{n_i} \to \mathbb{R}$ and collection of variables $x_i \in \mathbb{R}^{n_i}$ along with a matrix A_i such that $A_i x_i$ is the amount of power generated by the asset at the various time steps. A vector $d \in \mathbb{R}^T$, with T the total number of time steps, each of length Δt (hours) is the system wide load.

The abstract problem reads:

$$\begin{aligned} \min_{x_1,\ldots,x_m} \quad & \sum_{i=1}^{m} c_i(x_i) \\ \text{s.t.} \quad & x_i \in X_i, \\ & \sum_{i=1}^{m} A_i x_i \geq d. \end{aligned}$$

It is very popular to cast this unit-commitment problem as a MILP. This is by no means the only path, nor necessarily the most desirable one. It is however fitting for this course.

This project will thus consist of a writing down and implementation of the resulting optimization problem as well as a resolution / testing of the code.

5.2 Thermal plants

Consider the situation of a (thermal) power plant producing over a time horizon T at each time step p_t, $t = 1, ..., T$ (MW) of active power. Each time step has a duration Δt in hours. The power plant has a proportional cost c_t in €/ MWh ; Power levels are subject to a minimal and maximum power output level when running $p_t^{\texttt{min}}, p_t^{\texttt{max}}$ respectively. Furthermore the power levels are subject to gradient restrictions, i.e., adjacent power levels should not exceed g_t, where g_t is in MW/h; When the plant is online, it needs to remain so for a minimum of τ_+ time steps. When the plant is offline it needs to remain so for a total of τ_- time steps.

Thermal plants moreover have a startup cost s_t € that has to be paid

5.3 Cascading reservoir systems

A cascading reservoir system can be understood as a directed graph $G = (V, A)$, wherein each vertex $v \in V$ represents a reservoir with initial volume V_0 (m^3), minimal volume $V_t^{\texttt{min}}$ and maximal volume $V_t^{\texttt{max}}$ in m^3. Each reservoir also receives inflows a_t in m^3/s. The arcs are directed from uphill to downhill and represent turbines (going with the direction) or pumps (going against the direction).

Each turbine comes with a flow rate f_t in m^3/s as well as ramping conditions g in $(m^3/s)/h$. The flow rate are subject to bounds $\underline{f}_t, \overline{f}_t$ and moreover we are given the following form of the hydro production function:

$$p(f) = \min_{j \in J} p_j + \langle \rho_j, f - f_j \rangle \,,$$

with for $j \in J$: f_j (m^3/s), p_j (MW) a fixed collection of points. The resulting power output of a turbine should also be subject to bounds $\underline{p}, \overline{p}$.

Each pump likewise comes with a flow rate (negative) but with a unique linear "HPF" : $p(f) = \rho f$. Note that physically pumps require power to pump water uphill. Evidently this must go at an overall loss (think about why?).

5.4 Description of the task at hand

In order to tackle the project, the first step consists of making precise the sets of constraints X_i for the various thermal and hydro assets. To this end observe that i as hydro is concerned relates to a cascading reservoir system as a whole. Once these constraints are written down, write the full UC problem as a MILP: think carefully about the proper formulation.

The next step consists in writing a computer program that implements the model. It will have to read data from some source, populating the model, build the model, solve the model, output the solution to some human readable format.

The proper way to structure the program is to clearly distinguish between the various steps. Data reading and curing should not be done during the building of the model phase. Once the model is built, it should be exportable to some classic format (MPS of LP files for instance), making it solver agnostic. The solving should be done with some solver (CPLEX, Gurobi, or other available), possibly playing with parameters of the solver. The data output phase should provide some display of the solution in an understandable way. In this case, we should at least be able to see how much power is generated by each asset at each time step. For cascading reservoir systems we may wish to see the flow of water in between reservoirs, or how the volumes in each reservoir evolve over time.

In terms of programming language, Python PULP, Pyomo or Julia JUMP are likely good suggestions.

5.5 Data

Download some thermal unit-commitment test data at `https://gitlab.com/smspp/ucblock/-/tree/develop/netCDF_files/UC_Data/T-Ramp`. The netcdf files should be transformed into clear text through ncdump.

Bibliography

[1] J.F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

[2] R. G. Bland. New finite pivoting rules for the simplex method. *Mathematics of Operations Research*, 2(2):103–107, 1977.

[3] N. Karmakar. A new polynomial time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

[4] J. E. Kelley. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

[5] J. Luedtke and S. Ahmed. A sample approximation approach for optimization with probabilistic constraints. *SIAM Journal on Optimization*, 19:674–699, 2008.

[6] J. F. Maurras, K. Truemper, and M. Akgül. Polynomial algorithms for a class of linear programs. *Mathematical Programming*, 21:121–136, 1981.

[7] C. Roos, T. Terlaky, and J-P. Vial. *Interior Point Methods for Linear Optimization*. Springer-Verlag New York, 2nd edition, 2005.

[8] W. van Ackooij. A discussion of probability functions and constraints from a variational perspective. *Set-Valued and Variational Analysis*, 28(4):585–609, 2020.

[9] W. van Ackooij, I. Danti Lopez, A. Frangioni, F. Lacalandra, and M. Tahanan. Large-scale unit commitment under uncertainty: an updated literature survey. *Annals of Operations Research*, 271(1):11–85, 2018.

[10] W. van Ackooij, A. Frangioni, and W. de Oliveira. Inexact stabilized Benders' decomposition approaches: with application to chance-constrained problems with finite support. *Computational Optimization And Applications*, 65(3):637–669, 2016.