Optimisation *

Zhiyuan Bai

Compiled on October 2, 2020

This document serves as a set of revision materials for the Cambridge Mathematical Tripos Part IB course *Optimisation* in Easter 2020. However, despite its primary focus, readers should note that it is NOT a verbatim recall of the lectures, since the author might have made further amendments in the content. Therefore, there should always be provisions for errors and typos while this material is being used.

Contents

*Based on the lectures under the same name taught by Dr. M. R. Tehranchi in Easter 2020.

0 Introduction

Our aim for this course is to explore optimisation problems, typically minimising a function $f(x)$ with $f : \mathbb{R}^n \to \mathbb{R}$ the objective function subject to the regional constraint $g(x) = b, x \in X$ where $X \subset \mathbb{R}^n$ defines a regional constraint, and $g : \mathbb{R}^n \to \mathbb{R}^m$ defines m functional constraints.
A feasible solution to the problem is any $x \in X$ such that $g(x) = b$, so a feasible solution is just one that satisfy the constraint of the problem. An optimal solution is a feasible solution $x^\star$ such that $f(x^\star) \leq f(x)$ for all feasible x. The problem itself is feasible if there exists at least one feasible solution, and it is bounded if $\inf\{f(x) : g(x) = b, x \in X\} \in \mathbb{R}$, which is the main focus of us.
Notice that it makes no difference if we consider the maximisation instead of minimisation by putting appropriate minus signs in the definitions.

1 Convex Optimisation

Consider the simple problem of minimizing $f(x)$ subject to $a \leq x \leq b$ in the case where $f : \mathbb{R} \to \mathbb{R}$ is twice differentiable.

Theorem 1.1. *If $x^\star \notin \{a, b\}$ is an optimal solution of the problem, then $f'(x^\star) = 0$.*

Proof. For sufficiently small $\epsilon > 0$ we can have both $x^\star - \epsilon$ and $x^\star + \epsilon$ are feasible (a fancy way of saying $(x^\star - \epsilon, x^\star + \epsilon) \subset [a, b]$), then by optimality

$$\frac{f(x^\star) - f(x^\star - \epsilon)}{\epsilon} \leq 0 \leq \frac{f(x^\star + \epsilon) - f(x)}{\epsilon}$$

Sending $\epsilon \to 0$ gives the result. □

As well known, there is a partial converse to this theorem.

Theorem 1.2 (Sufficient Condition for Optimality)**.** *Suppose $x^\star$ is feasible and $f'(x^\star) = 0$. If $f'' \geq 0$ for all feasible x, then $x^\star$ is optimal.*

Proof. By Taylor's Theorem (used correctly!), let x be a feasible solution, then

$$f(x) = f(x^\star) + f'(x^\star)(x - x^\star) + \frac{1}{2} f''(\xi)(x - x^\star)^2$$

for some ξ in between x and $x^\star$. As $f'(x^\star) = 0$ and $f''(\xi) \geq 0$ by assumption, $f(x) \geq f(x^\star)$. □

The one-dimensional optimisation problems are widely studied and there is not much point to continue talking about it. Let's move on to higher dimensions. As it turns out, convexity helps a lot on problems without a functional constraint.

1.1 General Ideas on Convexity

Notationally, for a differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, we write Df as its gradient and, when f is twice differentiable, $D^2 f$ as its Hessian. Recall that the Hessian is symmetric at x when all the second order partial derivatives are continuous at x.

Definition 1.1. A set $X \subset \mathbb{R}^n$ is convex if for every pair of points $x, y \in X$ and number $0 < p < 1$ we have $px + (1-p)y \in X$.
Loosely speaking, the line segment joining x and y is still in X.

One should note that our solution to the previous one dimensional optimisation problem somehow depends on the convexness of the contraint set. [1]

Definition 1.2. Let $X \subset \mathbb{R}^n$ be convex. A function $f : X \to \mathbb{R}$ is convex if for every pair of points $x, y \in X$ and $p \in (0, 1)$,

$$f(px + (1-p)y) \leq pf(x) + (1-p)f(y)$$

Respectively, a function is concave if its negative is convex. This definition is intuitive in one dimension, but might not be that easy to illustrate in higher dimensions on first sight. We therefore need the following reformulation of the idea.

Theorem 1.3 (Supporting Hyperplane)**.** *Let $X \subset \mathbb{R}^n$ be convex. The function $f : X \to \mathbb{R}$ is convex iff for every $x \in X$ there exists a vector $\lambda(x) \in \mathbb{R}^n$ such that*

$$f(y) - f(x) \geq \lambda(x)^\top (y - x)$$

for all $y \in X$.

Remark. If f is also differentiable, then we can take $\lambda(x) = Df(x)$.

One can picture it as if the hyperplane produced by $\lambda(x)$ touches and supports the high-dimensional object that is the graph of the function. This is the origin of the name.
We will prove the case when f is differentiable, but the general case is also true.

Proof. Almost immediate but let's write this out. First suppose $\lambda(x)$ exists for all x. Fix $y, z \in X$ and $p \in (0, 1)$ and let $x = py + (1-p)z$, then

$$f(y) - f(x) \geq \lambda(x)^\top (y - x), f(z) - f(x) \geq \lambda(x)^\top (z - x)$$

So

$$pf(y) + (1-p)f(z) \geq \lambda(x)^\top (py + (1-p)z - x) + f(x) = f(x)$$

Conversely, if f is convex, we can just take $\lambda(x) = Df(x)$. □

Hence, for the problem of minimising $f(x)$ subject to $x \in X$ where X is convex and $f : X \to \mathbb{R}$ is differentiable, we have

Theorem 1.4. *Suppose $x^\star$ is feasible and $Df(x^\star) = 0$. If f is convex, then $x^\star$ is optimal.*

[1] Of course, even if it is not, we can sometimes produce similar results as well

Proof. Suppose x is feasible, then by convexity of f,

$$f(x) - f(x^\star) \geq (x - x^\star)^\top Df(x^\star) = 0$$

which is what we wanted. □

Definition 1.3. A symmetric $n \times n$ matrix A is non-negative definite if for every $x \in \mathbb{R}^n$ we have $x^\top Ax \geq 0$.

Theorem 1.5. *Let $X \subset \mathbb{R}^n$ be convex and suppose $f : X \to \mathbb{R}^n$ is C^2. If $D^2 f$ is nonnegative definite for all x, then f is convex.*

Proof. For any $x, y \in X$ we have, by Taylor's Theorem that

$$f(y) = f(x) + (y - x)^\top Df(x) + \frac{1}{2}(y - x)^\top D^2 f(\xi)(y - x)$$

where $\xi = px + (1-p)y$ for some $p \in (0, 1)$. Therefore $f(y) - f(x) \geq Df(x)^\top (y - x)$, hence it is convex. □

Definition 1.4. Given a convex set $X \subset \mathbb{R}^n$, a function $f : X \to \mathbb{R}$ is strictly convex if for every $x, y \in X$ where $x \neq y$ and every number $p \in (0, 1)$ we have

$$f(px + (1 - p)y) < pf(x) + (1 - p)f(y)$$

Definition 1.5. An $n \times n$ matrix A is positive definite if $x^\top Ax > 0$ for all $x \in \mathbb{R}^n \setminus \{0\}$.

Theorem 1.6. *Suppose $f : X \to \mathbb{R}$ is twice differentiable. If $D^2 f$ is positive definite over all of X, then f is strictly convex.*

Proof. Same idea as in the ordinary convex case. □

Theorem 1.7 (Uniqueness of Optimal Solutions)**.** *Suppose $x^\star$ and $y^\star$ are optimal solutions to the problem of minimising $f(x)$ subject to $x \in X$ with X convex and f strictly convex, then $x^\star = y^\star$.*

Proof. If $x^\star \neq y^\star$, then let $z = (x^\star + y^\star)/2$, then

$$f(z) < \frac{1}{2}f(x^\star) + \frac{1}{2}f(y^\star) = f(x^\star)$$

Contradiction. □

Definition 1.6. A function $f : X \to \mathbb{R}$ is strongly convex if there exists a constant $m > 0$ such that the function $x \mapsto f(x) - m\|x\|^2/2$ is convex.

This defintion is certainly strange and unintuitive, but as we proceed, we will unveil the reason to introduce this notion.

Theorem 1.8. *Suppose f is twice differentiable. Then f is strongly convex if there exists $m > 0$ such that for all $x \in X$ the matrix $D^2 f(x) - mI$ is non-negative definite. Equivalently, $z^\top D^2 f(x) z \geq m\|z\|^2$ for all $z \in \mathbb{R}^n$.*

Proof. Obvious. □

One can check that strongly convex functions are also strictly convex.

Theorem 1.9 (Existence of an Optimal Solution)**.** *Suppose $X \subset \mathbb{R}^n$ is closed and f is strongly convex, then there exists an optimal solution to the problem of minimising $f(x)$ subject to $x \in X$.*

Proof. Let $g(x) = f(x) - m\|x\|^2/2$ where $m > 0$ such that g is convex, then there is a vector $\lambda = \lambda(0)$ such that

$$g(x) \geq g(0) + \lambda^\top x \geq g(0) - \|\lambda\|\|x\|$$

by Cauchy-Schwartz. So for $\|x\| > R = 2\|\lambda\|/m$ we have $f(x) > f(0)$ by rearranging. So our problem reduces to minimising $f(x)$ subject to $x \in X \cap B$ where B is the closed ball of radius R. But $X \cap B$ is closed and bounded, hence is compact, therefore f attains a minimum somewhere in it, which is the optimal sulution we want. □

Theorem 1.10 (Gradient Lower Bound)**.** *Suppose $f : X \to \mathbb{R}$ is differentiable and strongly convex with constant $m > 0$, then*

$$\|Df(x)\|^2 \geq 2m(f(x) - f(y))$$

for any $x, y \in X$.

Proof. From the Theorem 1.3,

$$f(y) - f(x) \geq (y - x)^\top Df(x) + \frac{m}{2}\|y - x\|^2$$

However, by simply conpleting the square, for any $b, z \in \mathbb{R}^n$,

$$b^\top z + \frac{m}{2}\|z\|^2 \geq -\frac{\|b\|^2}{2m}$$

Combining the inequalities give the solution. □

1.2 Computing the Optimal Solution

Our most familiar example of a convex set is $X = \mathbb{R}^n$. We want to find algorithms to compute the optimal solution in this case. Our first method of interest is called gradient descent. Suppose f is differentiable. The rate of change of f at a point $x \in \mathbb{R}^n$ in direction $u \in \mathbb{R}^n$ is

$$\lim_{t \to 0} \frac{f(x + tu) - f(x)}{t} = u^\top Df(x)$$

which attains minimum when u is in the direction $-Df(x)$ by Cauchy-Schwartz. So our algorithm will be as follows: Start with an initial guess $x_0 \in \mathbb{R}^n$ and pick a step suze $t > 0$. For every $k \geq 0$, we use the iteration

$$x_{k+1} = x_k - tDf(x_k)$$

to refine our guess. Intuitively, this should get closer and closer to an optimal solution. This might or might not be true, but for a sufficiently nice function f we have the following theorem:

Theorem 1.11. *Suppose f is C^2 and there exists positive constants $M > m > 0$ such that $mI \preceq D^2 f(x) \preceq MI$ for all $x \in \mathbb{R}^n$. Here, two symmetric matrices A, B has $A \preceq B$ iff $x^\top Ax \leq x^\top Bx$ for every $x \in \mathbb{R}^n$. The lower bound, in particular, implies the strong convexity, hence existence and uniqueness of $x^\star$. Then the gradient algorithm with step size $t = 1/M$ has*

$$f(x_k) - f(x^\star) \leq \left(1 - \frac{m}{M}\right)^k (f(x_0) - f(x^\star))$$

Proof. Fix $x, y \in \mathbb{R}^n$, then by Taylor's Theorem, there is some $p \in (0,1)$ such that for $\xi = px + (1-p)y$ we have

$$\begin{aligned} f(y) &= f(x) + (y-x)^\top Df(x) + \frac{1}{2}(y-x)^\top D^2 f(\xi)(y-x) \\ &\leq f(x) + (y-x)^\top Df(x) + \frac{M}{2}\|y-x\|^2 \end{aligned}$$

Apply this inequality to $y = x_{k+1}, x = x_k$ gives

$$\begin{aligned} f(x_{k+1}) - f(x_k) &\leq (x_{k+1} - x_k)^\top Df(x_k) + \frac{M}{2}\|x_{k+1} - x_k\|^2 \\ &= \left(-t + \frac{M}{2}t^2\right)\|Df(x_k)\|^2 \\ &= -\frac{1}{2M}\|Df(x_k)\|^2 \\ &\leq -\frac{m}{M}(f(x_k) - f(x^\star)) \end{aligned}$$

Rearranging gives

$$f(x_{k+1}) - f(x_k) \leq \left(1 - \frac{m}{M}\right)(f(x_k) - f(x^\star))$$

Using this recursively gives the estimate we need. □

This is certainly a very nice method since the rate of convergence we arrive at looks quite promising. There is, of course, a better method similar to our familiar formualtion of Newton's method of finding root (since very roughly speaking we are finding a root of the derivative) but in high dimensions. To confuse people, we shall also call this Newton's method.
Suppose f is C^2 and let x_0 be an initial guess for the optimiser, then by Taylor's Theorem,

$$f(x) \approx f(x_0) + (x - x_0)Df(x_0) + \frac{1}{2}(x - x_0)^\top D^2 f(x_0)(x - x_0)$$

as $x \to x_0$ due to the continuity of $D^2 f$. Minimising the quadratic term on the right yields the approximation

$$x^\star \approx x_0 - (D^2 f(x_0))^{-1} Df(x_0)$$

Formally, we are referring to the recursion

$$x_{k+1} = x_k - (D^2 f(x_0))^{-1} Df(x_0)$$

As before, we want to know if it works.

Definition 1.7. Let A be an $n \times n$ matrix. We define the matrix norm $\|A\|$ as

$$\|A\| = \inf\{a \geq 0 : \forall z \in \mathbb{R}^n, \|Az\| \leq a\|z\|\} = \sup_{\|z\|=1} \|Az\|$$

In particular, if A is nonnegative definite, then $\|A\|$ is the smallest eigenvalue of A.

Theorem 1.12. *Suppose f is twice differentiable and there exists constants $m, L > 0$ with $D^2 f(x) \succeq mI$ (so f is strongly convex) and*

$$\forall x, y \in \mathbb{R}^n, \|D^2 f(x) - D^2 f(y)\| \leq L\|x - y\|$$

(so $D^2 f$ is L-Lipschitz), then Newton's method guarantees that

$$f(x_k) - f(x^\star) \leq \frac{2m^3}{L^2} \left(\frac{L}{2m^2} \|Df(x_0)\| \right)^{2^{k+1}}$$

Note that this convergence rate is way quicker than using gradient descent.

Proof. Denote $\Delta x_k = x_{k+1} - x_k = -(D^2 f(x_0))^{-1} Df(x_0)$, then we have

$$\begin{aligned}
\|Df(x_{k+1})\| &= \|Df(x_{k+1}) - Df(x_k) - D^2 f(x_k)\Delta x_k\| \\
&= \left\| \int_0^1 (D^2 f(x_k + t\Delta x_k) - D^2(x_k))\Delta x_k \, \mathrm{d}t \right\| \\
&\leq \int_0^1 \|(D^2 f(x_k + t\Delta x_k) - D^2(x_k))\Delta x_k\| \, \mathrm{d}t \\
&\leq L\|\Delta x_k\|^2 \int_0^1 t \, \mathrm{d}t \\
&= \frac{1}{2} L \|(D^2 f(x_0))^{-1} Df(x_0)\|^2 \\
&\leq \frac{L}{2m^2} \|Df(x_k)\|^2
\end{aligned}$$

Note here that we have used the fact that $A \succeq mI \implies \|A^{-1}\| \leq 1/m$ which is quite obvious. Applying this inequality recursively yields

$$\sqrt{2m(f(x_k) - f(x^\star))} \leq \|Df(x_k)\| \leq \frac{2m^2}{L} \left(\frac{L}{2m^2} \|Df(x_0)\| \right)^{2^k}$$

by Theorem 1.10. Rearranging shows what we want. □

2 The Lagrangian

2.1 The Sufficiency Theorem

We now consider the general case of the constrained optimisation, without explicitly appealing to the convexity assumption. Suppose we want to minimise $f(x)$ subject to $g(x) = b, x \in X$, to deal with the functional constraint, we introduce the Lagrangian of the problem:

Definition 2.1 (Lagrangian)**.** The Lagrangian $L : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ of the problem is defined by

$$L(x, \lambda) = f(x) + \lambda^\top (b - g(x))$$

For $\lambda = (\lambda_1, \ldots, \lambda_m)^\top$, the components λ_i constitutes the Lagrange multiplier.

A nice theorem associated with this notion is the following:

Theorem 2.1 (The Lagrange Sufficiency Theorem)**.** *Let $x^\star$ be feasible for the problem. Suppose there exists a $\lambda^\star \in \mathbb{R}^m$ such that*

$$\forall x \in X, L(x^\star, \lambda^\star) \leq L(x, \lambda^\star)$$

Then $x^\star$ is optimal.

Proof. For any feasible x and any $\lambda \in \mathbb{R}^m$, we have

$$L(x, \lambda) = f(x) + \lambda^\top (b - g(x)) = f(x)$$

So our assumption gives, for any feasible x,

$$f(x^\star) = L(x^\star, \lambda^\star) \leq L(x, \lambda^\star) = f(x)$$

as desired. □

Example 2.1. Consider the problem of minimising $x_1^2 + 3x_2^2$ subject to $4x_1 + x_2 = 7$. We claim that $(12/7, 1/7)$ is optimal. The Lagrangian has the expression

$$L(x_1, x_2, \lambda) = x_1^2 + 3x_2^2 + \lambda(7 - 4x_1 - x_2)$$

Take $\lambda = 6/7$, we have

$$L(x_1, x_2, 6/7) = (x_1 - 12/7)^2 + 3(x_2 - 1/7)^2 + 3 \geq 3 = L(12/7, 1/7, 6/7)$$

So $(12/7, 1/7)$ is optimal by the preceding theorem.

Note that our choice of multiplier $\lambda^\star = 6/7$ is kind of "out of nowhere". It makes one wonder if there is a systematic method to guarantee to produce a $\lambda^\star$ that works. Sadly, it is not always possible to find a multiplier that works as a certificate of optimality, let alone a sufficient algorithm to compute one. However, inspired by the theorem, we can come up with a way to compute $\lambda^\star$ if the problem behaves sufficiently nice.
If $x^\star$ and $\lambda^\star$ do exist, we must have

$$\inf_{x \in X} L(x, \lambda^\star) = f(x^\star) > -\infty$$

So to calculate $\lambda^\star$, we first identify the possible candidates

$$\Lambda = \{\lambda \in \mathbb{R}^n : \inf_{x \in X} L(x, \lambda) > -\infty\}$$

Then, for each $\lambda \in \Lambda$, we consider the probelm of minimising $L(x, \lambda)$ subject to $x \in X$. This is hopefully easiler since the functional constraint has been removed (with, of course, the trade-off of having to consider the problem for a wide range of functions). Denote then minimiser as $x(\lambda)$. We now find a $\lambda^\star \in \Lambda$ such that $x(\lambda^\star)$ is feasible. In general, it might not be possible to go through all these steps, but if it is possible, then $x^\star = x(\lambda^\star)$ would be guaranteed to be the optimal solution by the theorem.

Example 2.2 (Maximum Likelihood Estimator of the Multinomial Distribution)**.** Given constants $n_1, \ldots, n_k > 0$, we consider the problem to maximise $\sum_i n_i \log p_i$ subject to $\sum_i p_i = 1, \forall i, p_i > 0$. The Lagrangian is

$$L(p, \lambda) = \lambda + \sum_{i=1}^{k} (n_i \log p_i - \lambda p_i)$$

Then easily $\Lambda = \mathbb{R}_{>0}$. Now we have

$$\frac{\partial L}{\partial p_i} = \frac{n_i}{p_i} - \lambda = 0 \implies p_i(\lambda) = \frac{n_i}{\lambda}$$

But also

$$D^2 L = \begin{pmatrix} -n_1/p_1^2 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & -n_k/p_k^2 \end{pmatrix}$$

which is negative definite, so we have found the maximum. The constraint $\sum_i p_i = 1$ gives $\lambda = \sum_i n_i$, so

$$p_i^\star = \frac{n_i}{\sum_j n_j}$$

is optimal.

2.2 Inequality Constraints

We can consider a somewhat more general version of the optimisation problem, where we replace the functional constraints by their inequality analog. For vectors x, y, we write $x \leq y$ if $x_i \leq y_i$ for each i.
Let $f : \mathbb{R}^n \to \mathbb{R}, g : \mathbb{R}^n \to \mathbb{R}^m, X \in \mathbb{R}^n$. We want to consider the problems to minimise $f(x)$ subject to $g(x) \leq b, x \in X$. Notice that we can actually rewrite this problem to an equality form by introducing a slack variable $z \in \mathbb{R}^m$ by considering the minimisation of $f(x)$ subject to $g(x) + z = b, x \in X, z \geq 0$.
Then, the Lagrangian of the modified problem would be

$$L(x, z, \lambda) = f(x) + \lambda^\top (b - g(x) - z) = f(x) + \lambda^\top (b - g(x)) - \lambda^\top z$$

Note that if some of λ is positive then we can easily send L to $-\infty$, so $\Lambda \subset \{\lambda \in \mathbb{R}^m : \lambda \leq 0\}$. So if we want to use the Lagrange multiplier method, then a sign constraint is further imposed on λ. In particular,

$$\Lambda = \{\lambda \in \mathbb{R}^m : \lambda \leq 0, \inf_{x \in X} (f(x) + \lambda^\top (b - g(x))) > -\infty\}$$

Now we proceed to the second step of using Lagrange multiplier. Note that for $\lambda \leq 0$, we have $\inf_{z \geq 0}(-\lambda^\top z) = 0$, so the optimal $z = z(\lambda)$ would satisfy the complementary slackness condition, i.e. $\lambda^\top z = 0$. So if the i^{th} Lagrange multiplier is nonzero, then $z_i = 0$, so the i^{th} functional constraint is tight, i.e. hold with equality. If the i^{th} functional constraint is not tight, then $z_i > 0$, thus it must be the case that $\lambda_i = 0$. The rest of the method can then be carried on as usual.

Example 2.3. Consider the minimisation of $x_1 - 3x_2$ subject to $x_1^2 + x_2^2 \leq 4$ and $x_1 + x_2 \leq 2$. We introduce the slack variables $z_1, z_2 \geq 0$, by which we can transform the problem to the minimisation of $x_1 - 3x_2$ subject to

$$\begin{cases} x_1^2 + x_2^2 + z_1 = 4 \\ x_1 + x_2 + z_2 = 2 \\ z_1, z_2 \geq 0 \end{cases}$$

So the Lagrangian is

$$L = x_1 - 3x_3 + \lambda_1(4 - x_1^2 - x_2^2 - z_1) + \lambda_2(2 - x_1 - x_2 - z_2), \lambda_1, \lambda_2 \leq 0$$

Note that

$$D^2 L = \begin{pmatrix} -2\lambda_1 & 0 \\ 0 & -2\lambda_1 \end{pmatrix} \succeq 0$$

Hence we only need $\partial L/\partial x_1 = \partial L/\partial x_2 = 0$, whcih yields

$$\begin{cases} 1 - 2\lambda_1 x_1 - \lambda_2 = 0 \\ -3 - 2\lambda_1 x_2 - \lambda_2 = 0 \end{cases}$$

If $\lambda_1 = 0$, then $\lambda_2 = 1$ and $\lambda_2 = -3$, contradiction. So $\lambda_1 < 0$. if $\lambda_2 < 0$, then by complementary slackness, $z = 0$, therefore

$$\begin{cases} 1 - 2\lambda_1 x_1 - \lambda_2 = 0 \\ -3 - 2\lambda_1 x_2 - \lambda_2 = 0 \\ x_1^2 + x_2^2 = 4 \\ x_1 + x_2 = 2 \end{cases}$$

Solving this equation gives $(x_1, x_2) \in \{(2, 0), (0, 2)\}$, so $(x_1, x_2, \lambda_1, \lambda_2)$ is either $(2, 0, 1, -3)$ or $(0, 2, -1, 1)$, but both violates the sign constraint, which gives contradiction again. Hence $\lambda_2 = 0$. By complementary slackness $z_1 = 0$, so the first functional constraint is tight, hence

$$\begin{cases} 1 - 2\lambda_1 x_1 = 0 \\ -3 - 2\lambda_1 x_2 = 0 \\ x_1^2 + x_2^2 = 4 \end{cases}$$

Solve to get $\lambda_1 = -\sqrt{10}/4$ and

$$(x_1, x_2) = \left(-\sqrt{\frac{2}{5}}, 3\sqrt{\frac{2}{5}}\right)$$

which is feasible hence optimal.

While we can solve it, it is always good to know beforehand that this method will work. This brings us to the next topic on the necessity condition.

2.3 The Necessity Theorem

Let's go back to the problem of minimising $f(x)$ subject to $g(x) = b, x \in X$. Denote the Lagrangian by

$$L(x, \lambda) = f(x) + \lambda^\top (b - g(x))$$

We have already established Theorem 2.1 saying that a feasible $x^\star$ is optimal if there exists some $\lambda^\star$ with $L(x^\star, \lambda^\star) \leq L(x, \lambda^\star)$ for any $x \in X$. In particular, we have

$$\inf_{x \in X, g(x)=b} f(x) = f(x^\star) = \inf_{x \in X} L(x, \lambda^\star)$$

We, of course, want some useful statement on the converse of this statement.

Theorem 2.2 (Lagrangian Necessity)**.** *For $b \in \mathbb{R}^n$, let*

$$\phi(b) = \inf_{x \in X, g(x)=b} f(x)$$

If ϕ is defined and convex, then there is a $\lambda = \lambda(b)$ with

$$\phi(b) = \inf_{x \in X, g(x)=b} f(x) = \inf_{x \in X} L(x, \lambda(b))$$

Furthermore, if ϕ is differentiable, then $\lambda(b)$ can be taken to be $D\phi(b)$.

Definition 2.2. Such a function $\phi(b)$ is called the value function of the family of the problems of minimising $f(x)$ subject to $g(x) = b, x \in X$.

Proof. Suppose ϕ is convex and fix $b \in \mathbb{R}^m$, then by Theorem 1.3, there is some $\lambda \in \mathbb{R}^m$ (which can be taken to be $D\phi(b)$) such that

$$\begin{aligned}
\phi(b) &= \inf_{c \in \mathbb{R}^m} (\phi(c) + \lambda^\top (b - c)) \\
&= \inf_{c \in \mathbb{R}^m} \inf_{x \in X, g(x)=c} (f(x) + \lambda^\top (b - c)) \\
&= \inf_{c \in \mathbb{R}^m} \inf_{x \in X, g(x)=c} (f(x) + \lambda^\top (b - g(x))) \\
&= \inf_{x \in X} L(x, \lambda)
\end{aligned}$$

which is just what we wanted. □

Remark. By adding slack variables

$$\inf_{x \in X, g(x) \leq b} f(x) = \inf_{x \in X, z \geq 0, g(x)+z=b} f(x)$$

We can see that the theorem can still hold when the functional constraint is an equality.

To apply this theorem, we can require helps on determining the convexity of ϕ.

Theorem 2.3. *Suppose X is convex, f is convex, the functional constraint is $g(x) \leq b$ and g_j is convex for all $1 \leq j \leq m$, then ϕ is convex.*

Proof. Exercise. □

2.4 Shadow Prices

We can view the optimisation problem in some real-life economical setting, which gives intuition to the construction of the Lagrangian.
Consider a factory owner who makes n different products out of m raw materials. He needs to choose amount x_i to make of the i^{th} product for each $i = 1, \dots, n$. Given a vector of amounts $x = (x_1, \dots, x_n)^\top$ of products to manufacture, the factory requires the amount $g_j(x)$ of the j^{th} raw material for $j = 1, \dots, m$, and the amount of available j^{th} raw material available is b_j. The only nonnegative amounts of products can be produced and given the amounts x of products, the profit earned is $f(x)$.
Obviously, what the factory owner wants is to maximise $f(x)$ subject to $g(x) \leq b, x \geq 0$. We let $\phi(b)$ be the maximised profit as a function of the amount of raw materials.
To solve this problem, we can consider a fictional world where there is a market for the raw material with the unit price of the j^{th} raw material being λ_j, then the total profit is exactly

$$L(x, \lambda) = f(x) + \lambda^\top (b - g(x))$$

So the problem has transformed into the maximisation problem of the Lagrangian. For each price vector λ, the optimal amount of products is the vector $x(\lambda)$. We can now find a price vector $\lambda^\star$ such that $x^\star = x(\lambda)$ is feasible. Such a $\lambda^\star$ is called the shadow price vector of the raw materials. This is the prices of the raw materials such that the maximisation problem makes no difference on whether there is such a fictional market.
The supporting hyperplane can also have an economical interpretation. Suppose the fictional market of raw materials suddenly appears, then the factory owner would buy the basket $(\epsilon_1, \dots, \epsilon_m)^\top$ of raw materials if $\phi(b+\epsilon) - \phi(b) \geq \lambda^\top \epsilon$, so when ϵ is small,

$$\phi(b + \epsilon) - \phi(b) = D\phi(b)^\top \epsilon$$

So if $\lambda_j > \partial\phi/\partial b_j$, then the factory owner should not be buying any of that raw material. Hence the shadow price has to be $D\phi(b)$ as before.
To check that all these are consistent, we check the followings: The inequality constraint $g(x) \leq b$ gives a sign constraint $\lambda \geq 0$, which is consistent with the assumption that price should not be negative. Also, if the j^{th} constraint is not tight, then $g_j(x^\star) < b_j$, then the owner does not use up all the j^{th} raw material, so the shadow price has $\partial\phi/\partial b_j = 0$ as there is no extra profit in acquiring some more of that material. On the other hand, the j^{th} slack variable would be positive, hence $\lambda_j = 0$ by complementary slackness, which makes sense economically.

2.5 Duality

We can introduce the notion of a dual optimisation problem by a corresponding economic motivation. Consider the point of view of the seller of the raw materials who then buy the products produced by the factory. If the amount of finished products is x and the price of raw materials is λ, then the seller's profit would be the negative of the profit of factory owner, which is $f(x)+\lambda^\top(b-g(x))$.

So what the seller wants would be to minimise

$$\sup_{x \geq 0}(f(x) + \lambda^\top (b - g(x)))$$

Definition 2.3. Consider the primal problem of minimising $f(x)$ subject to $g(x) = b, x \in X$ which has Lagrangian

$$L(x, \lambda) = f(x) + \lambda^\top (b - g(x))$$

Then the set of feasible Lagrange multipliers is

$$\Lambda = \{\lambda \in \mathbb{R}^m : \inf_{x \in X} L(x, \lambda) > -\infty\}$$

Then the dual objective function is defined as

$$h : \Lambda \to \mathbb{R}, h(\lambda) = \inf_{x \in X} L(x, \lambda)$$

The dual problem of the primal problem is to maximise $h(\lambda)$ subject to $\lambda \in \Lambda$.

Remark. One should note that if one need to study the dual problem of a problem with inequality constraints, then we need to introduce the slack variables first. Also, we can formulate the dual problem to a maximisation problem analogously.

Theorem 2.4 (Weak Duality)**.** *With the definitions introduced above,*

$$\sup_{\lambda \in \Lambda} h(\lambda) \leq \inf_{x \in X, g(x) = b} f(x)$$

Proof. For any x feasible for the primal problem and λ feasible for the dual problem, then

$$h(\lambda) = \inf\{L(\xi, \lambda) : \xi \in X\} \leq L(x, \lambda) = f(x)$$

As $L(x, \lambda) = f(x)$ whenever x is feasible. □

So we can restate Theorem 2.1 in the following way instead: If $x^\star$ and $\lambda^\star$ are feasible for the primal and dual problems respectively, and $h(\lambda^\star) = f(x^\star)$, then $x^\star$ is optimal.

Remark. The difference

$$\inf_{x \in X, g(x) = b} f(x) - \sup_{\lambda \in \Lambda} h(\lambda)$$

is called the duality gap. It is in general nonnegative by Theorem 2.4. By Theorem 2.2, if the value function of the primal problem is convex, then there is some $\lambda^\star \in \Lambda$ such that

$$\inf_{x \in X, g(x) = b} f(x) = \inf_{x \in X} L(x, \lambda^\star) = h(\lambda^\star)$$

So the duality gap is zero. This is called *strong duality*.

2.6 Duality in Linear Programs

Definition 2.4. Linear Programs are problems of maximising $c^\top x$ subject to $Ax \leq b, x \geq 0$ where A is an $m \times n$ matrix, $b \in \mathbb{R}^m$ and $c \in \mathbb{R}^n$.

Of course, we can analyze the dual problem too. Introduce the slack variable $z \in \mathbb{R}^m$ and transform the problem to maximising $c^\top x$ subject to $Ax+z = b, x \geq 0, z \geq 0$. So the Lagrangian is

$$L(x, z, \lambda) = c^\top x - \lambda^\top (b - Ax - z) = b^\top \lambda + (c - A^\top \lambda)^\top x - \lambda^\top z$$

So

$$\Lambda = \{\lambda \in \mathbb{R}^m : \sup_{x,z \geq 0} L(x, z, \lambda) < \infty\} = \{\lambda \in \mathbb{R}^m : A^\top \lambda \geq c, \lambda \geq 0\}$$

Now the dual objective function is

$$\sup_{x \geq 0, z \geq 0} L(x, z, \lambda) = b^\top \lambda$$

for $\lambda \in \Lambda$. Hence the dual problem is to minimise $b^\top \lambda$ subject to $A^\top \lambda \geq c, \lambda \geq 0$. This looks just like our original problem.

2.7 The Barrier Method

Consider the optimisation problem of minimising a differentiable convex function $f : \mathbb{R}^n \to \mathbb{R}$ subject to $g(x) \leq b$, where $g : \mathbb{R}^n \to \mathbb{R}^m$ has every one of its components differentiable. To tackle it (computationally), we consider the family of unconstrained minimisation problems of

$$f(x) - \epsilon \sum_{i=1}^{m} \log(b_i - g_i(x))$$

for $\epsilon > 0$. Note that the constraint $g(x) < b$ is implicitly imposed due to the domain of log. The idea of it, as one might have already observed, is to add an extra term which blows up to positive infinity if the condition $g(x) < b$ is not satisfied. The reason why we use a term in the $\epsilon \log$ form is to smoothen the blow-up term for computational issues.

Theorem 2.5. *Suppose $x^\star$ is optimal for our original problem and x_ϵ is optimal for the problem shown above, then*

$$0 \leq f(x_\epsilon) - f(x^\star) \leq m\epsilon$$

To prove it, we shall introduce some preliminary notion. The Lagrangian for the original problem is $L(x, z, \lambda) = f(x) + \lambda^\top (b - z - g(x))$ for the slack variable $z \geq 0$ and the set of feasible Lagrange multipliers are

$$\Lambda = \{\lambda \in \mathbb{R}^m : \inf_{x \in \mathbb{R}^n, z \geq 0} L(x, z, \lambda) > -\infty\}$$

Claim. *Let $\lambda \in \mathbb{R}^m$ be such that $\lambda \leq 0$ and that there exists some $x_\lambda \in \mathbb{R}^n$ with*

$$Df(x_\lambda) = \sum_{i=1}^{m} \lambda_i Dg_i(x_\lambda)$$

s Then $\lambda \in \Lambda$ and the dual objective function for this problem is just

$$h(\lambda) = b^\top \lambda + f(x_\lambda) - \lambda^\top g(x_\lambda)$$

Proof. Recall that every convex differentiable function is minimised at any of its stationary points. Also note that $f(x) - \lambda^\top g(x)$ is convex. So for $x \in \mathbb{R}^n$ and $z \geq 0$ we have the bound

$$L(x, z, \lambda) = b^\top \lambda + f(x) - \lambda^\top g(x) - \lambda^\top z \geq b^\top \lambda + f(x_\lambda) - \lambda^\top g(x_\lambda)$$

Equality holds when $x = x_\lambda$ and (z, λ) satisfies complementary slackness. The rest follows. □

Proof of Theorem 2.5. Recall that a differentiable function on an open set has derivative 0 on any of its local minima. As x_ϵ is optimal to the modified problem, we necessarily have

$$Df(x_\epsilon) = -\epsilon \sum_{i=1}^{m} \frac{Dg_i(x_\epsilon)}{b_i - g(x_\epsilon)}$$

So we set, for $i = 1, \ldots, m$ we define $\lambda_i = -\epsilon/(b_i - g_i(x_\epsilon))$. Then

$$Df(x_\epsilon) = \sum_{i=1}^{m} \lambda_i Dg_i(x_\epsilon)$$

So by the preceding claim $\lambda \in \Lambda$. Therefore, by Theorem 2.4,

$$f(x_\epsilon) \geq f(x^\star) \geq h(\lambda) = f(x_\epsilon) + \lambda^\top (b - g(x_\epsilon)) = f(x_\epsilon) - m\epsilon$$

which is the bound we wanted. □

Remark. As f and g_i are convex for all i, the value function $\phi(b) = \inf\{f(x) : g(x) \leq b\}$ is convex. So by fixing b and assuming there exists an optimiser $x^\star$, Theorem 2.2 then tells us there is some $\lambda^\star$ with

$$L(x^\star, z^\star) = \inf\{L(x, z, \lambda^\star) : x \in \mathbb{R}^n, z \geq 0\}, z^\star = b - g(x^\star)$$

Theorem 2.5 inspires us to invent an algorithm to approach the optimal solution. Pick an initial guess $x_0 \in \mathbb{R}^n$ with $g(x_0) < b$ and some initial $\epsilon_0 > 0$. For $k \geq 0$, we solve the modified problem with $\epsilon = \epsilon_k$ approximately (using e.g. gradient descent or Newton's method) starting from x_k. Then x_{k+1} is defined to be the approximated optimal solution in this way, and reduce ϵ by $\epsilon_{k+1} = r\epsilon_k$ for some $r \in (0, 1)$.

3 Linear Programming

Recall that a linear program is the maximisation of $c^\top x$ subject to $Ax \leq b, x \geq 0$, where $c \in \mathbb{R}^n$, A is an $m \times n$ matrix and $b \in \mathbb{R}^m$. We also know its dual problem in the form of minimising $b^\top \lambda$ subject to $A^\top \lambda \geq c, \lambda \geq 0$.

3.1 The Fundamental Theorem

Theorem 3.1 (Fundamental Theorem of Linear Programming)**.** *Consider the primal problem stated. A vector $x^\star \in \mathbb{R}^n$ is optimal for the primal problem iff there is some vector $\lambda^\star \in \mathbb{R}^m$ such that:*
1. $x^\star$ is feasible for the primal problem.
2. $\lambda^\star$ is feasible for the dual problem.
3. $(b - Ax^\star)^\top \lambda^\star = 0 = (c - A^\top \lambda^\star)^\top x^\star$ (i.e. the complementary slackness).
Also, if these are true, then $\lambda^\star$ is optimal for the dual problem, and $c^\top x^\star = b^\top \lambda^\star$.

Proof. For the 'if' direction, note that if x, λ are feasible for the primal and dual problems, then by Theorem 2.4, $c^\top x \leq b^\top \lambda$. But if $x^\star$ and $\lambda^\star$ satisfy complementary slackness, $c^\top x^\star = b^\top \lambda^\star$, consequently $c^\top x \leq b^\top \lambda^\star \leq c^\top x^\star$ for all feasible x and $b^\top \lambda \geq c^\top x^\star = b^\top \lambda^\star$ for all feasible λ. Hence $x^\star$ and $\lambda^\star$ are optimal for their respective problems.
For the "only if" direction, note that the objective function $f(x) = c^\top x$ and constraint function $g(x) = Ax$ are linear, hence are both concave and convex at the same time. In particular, f is concave while g is convex. Hence the value function of the primal function is concave, therefore Theorem 2.2 guarantees the existence of $\lambda^\star$. □

3.2 Extreme Points

Consider the maximisation of $\psi(x)$ subject to $x \in X$ where X is convex and ψ is also convex. In particular, for any $p \in (0, 1)$,

$$\psi(px + (1-p)y) \leq p\psi(x) + (1-p)\psi(y) \leq \max\{\psi(x), \psi(y)\}$$

Hence the maximum of ψ on any segment occurs at one of the endpoints. Therefore it is sufficient to consider the maximisation problem in the set of points in X that is not contained properly in a line segment.

Definition 3.1. Let $X \subset \mathbb{R}^n$ be a convex set. A point $x \in X$ is an extreme point if the only case where there can be $y, z \in X, p \in (0, 1)$ with $x = py + (1-p)z$ is when $x = y = z$.

Extreme points are of course very useful in linear programs as linear functions are both concave and convex. The concavity tells us we can characterise the optimal solution via duality. As the set of feasible solutions is convex, the convexity tells us we only need to search for optimal solutions to the extreme points of the set of feasible solutions. In particular, if the linear program has an optimal solution, then it has a optimiser that is an extreme point.

3.3 Basic Feasible Solutions

As we have seen, it is helpful if we formulate linear programs in the form of inequality functional constraints. There is another form which we can write the linear programs in order to make it more convenient to optimise.

Definition 3.2. A linear program is in standard form if the constraint can be written as $Ax = b, x \geq 0$.

All linear programs can be put into the standard form. For instance, $Ax \leq b, x \geq 0$ can be transformed to $Ax + z = b, x, z \geq 0$ by our previous idea of slack variables. Simply $Ax = b$ can be modified to $A(x - y) = b, x, y \geq 0$. Combining the ideas, $Ax \leq b$ can be transformed to $A(x - y) + z, x, y, z \geq 0$.

Definition 3.3. Given an $m \times n$ matrix A with $n > m$ and $b \in \mathbb{R}^m$. A solution $x \in \mathbb{R}^n$ of the equation $Ax = b$ is called basic if at most m entries of x are nonzero, i.e. $x_i \neq 0$ holds for at most m different $i \in \{1, \ldots, n\}$.
If x is a basic solution and $x \geq 0$, it is called a basic feasible solution.

Now denote the set of feasible solutions by $C = \{x \geq 0 : Ax = b\}$, then we have

Theorem 3.2. *If x is an extreme point of C, then x is a basic feasible solution.*

Proof. We will prove the contrapositive. Let x be a point in C that is not basic, then there are at least $m+1$ indices $i_1, \ldots, i_r$ with $x_{i_k} > 0$. Then the set $\{A_{i_1}, \ldots, A_{i_r}\}$ of columns of the matrix A would be linearly dependent as $t \geq m+1$, therefore there exists $(w_1, \ldots, w_r) \neq 0$ such that $w_a A_{i_a} = 0$ (summation implied). We construct

$$z_i = \begin{cases} w_k, \text{ if } i = i_k \text{ for some } k = 1, \ldots, r \\ 0, \text{ otherwise} \end{cases}$$

Then $Az = 0$ and $z \neq 0$, so $A(x \pm \epsilon z) = Ax = b$ for any ϵ for any ϵ. As $x_{i_k} > 0$ for any k, it is possible to choose some $\epsilon > 0$ such that $x \pm \epsilon z \geq 0$. So $x \pm \epsilon z \in C$ and

$$x = \frac{1}{2}(x + \epsilon z) + \frac{1}{2}(x - \epsilon_z)$$

Hence x is not extreme. □

This theorem has a partial converse.

Theorem 3.3. *Suppose that every set of m columns of A is linearly independent and x is a basic feasible solution, then x is necessarily an extreme point of C.*

Proof. By definition, at most m many indices i satisfy $x_i > 0$. Suppose there is some $y, z \in C$ and $p \in (0,1)$ such that $x = py + (1-p)z$, then $y, z \geq 0$, so whenever $x_i = 0$ we have $y_i = z_i = 0$. Thus there are at least $n - m$ indices i satisfying $y_i = z_i = 0$. But also $Ay = b = Az$, so

$$0 = A(y - z) = \sum_i w_i A_i$$

where $w = y - z$. But at most m entries of w are nonzero while any set of m columns of A is linearly independent, whence we conclude $w = 0$. It follows that $x = y = z$. □

In conclusion, we can find the optimal solution of a linear program by checking each of the basic feasible solutions, whcih are easier to find and, more importantly, finite.

3.4 Algorithms for Linear Programs

Consider a linear program in standard form of maximising $c^\top x$ subject to $Ax = b, x \geq 0$ where $c \in \mathbb{R}^n, b \in \mathbb{R}^m$ and A is an $m \times n$ matrix. Assume that $n > m$ and every set of m columns of A is linearly independent, and that the problem is non-degenerate, i.e. every basic feasible solution x has exactly m indices i with $x_i \neq 0$. Our goal is to find a mechanical method of finding the optimal solution to the problem.
A naïve method to do this is to check all extreme points of C. But we also know that extreme points are precisely the basic feasible solutions. Thus we can roll out an algorithm as follows:

First, fix $B \subset \{1, \ldots, n\}$ with $|B| = m$ write and $N = \{1, \ldots, n\} \setminus B$. Such a B is called a basis of the solution. Suppose $B = \{i_1, \ldots, i_m\}$, then we define $A_B = (A_{i_1} \ldots A_{i_m})$ which is an invertible matrix by assumption. For any $x \in \mathbb{R}^n$, denote $x_B = (x_{i_1}, \ldots, x_{i_m})^\top$. The quantities c_B, A_N, x_N, c_N are defined similarly.
Our aim is to find a basic solution of $Ax = b$. We can write the problem instead in the form $A_B x_B + A_N x_N = b$. Set $x_N = 0$ shall yield $x_B = A_B^{-1} b$ as A_B is invertible. By a rearrangement of coordinate we can write the solution as

$$x = \begin{pmatrix} x_B \\ x_N \end{pmatrix} = \begin{pmatrix} A_B^{-1} b \\ 0 \end{pmatrix}$$

which is basic. Now x is feasible iff $A_B^{-1} b \geq 0$ (which can be checked). Assuming x is a basic feasible solution, then we can just compute the objective function $c^\top x = c_B^\top x_B$. Repeat this procedure for all possible choices of B gives a list of all basic feasible solutions and the value of objective function evaluated there. This is a finite set, so we can find its maximiser which is the solution to the problem.
There are some issues with this approach. Firstly, if n is large, the number of choices of B, that is $\binom{n}{m}$, would grow very fast, making it more computationally complex. Also, this method assumed the existence of maximiser which is not always the case. Our idea algorithm should certainly detect it when a maximiser does not exist, i.e. the problem is unbounded.
There is a better method to accomplish what we wanted. We also use the idea of computing the basic feasible solutions like above, but we will try not to evaluate all candidate solutions. By calculation we can show that the dual problem is to minimise $b^\top \lambda$ subject to $A^\top \lambda \geq c$. So by the fundamental theorem, a feasible $x^\star$ is optimal for our primal problem iff there is a feasible (in the dual problem) Lagrange multiplier $\lambda^\star \in \mathbb{R}^m$ such that complementary slackness $(c - A^\top \lambda^\star)^\top x^\star = 0$ is satisfied.
Consider a basic feasible solution $x = (A_B^{-1} b, 0)^\top$, then if it is optimal, then there is some λ satisfying complementary slackness

$$0 = (c - A^\top \lambda)^\top x = (c_B - A_B^\top \lambda)^\top x_B$$

But no index of x_B is zero by assumption, hence $\lambda = (A_B^\top)^{-1} c_B$. So we can roll out our algorithm in the same way as before, but each time we check as well the feasibility of λ. We can stop if such λ is feasible, because by the fundamental theorem we have already found the optimal solution. We also know when the problem is unbounded, that is when none of the basic feasible solution gives a feasible λ.

3.5 The Simplex Algorithm

A third way, which is known as the simplex algorithm, is a yet faster method. So far our algorithms depend on iterating all candidates for basic feasible solutions randomly, which inspires us to find the solutions in a particular order to speed up the process. In particular, if our current basic feasible solution does not work, we will want to find the next basic feasible solution that increases the objective function.
We start with an initial basic feasible solution x_0 whose components with indices

in $B_0 \subset \{1, \ldots, n\}$ are nonzero. Write $N_0 = \{1, \ldots, n\} \setminus B_0$. We can associate a Lagrange multiplier λ_0 by complementary slackness, i.e. one that satisfies $(c - A^\top \lambda_0)_{B_0} = 0$. In particular, $(c - A^\top \lambda_0)^\top x_0 = 0$, so $c^\top x_0 = b^\top \lambda_0$. Let $\mu_0 = c - A^\top \lambda_0$. If $\mu_0 \leq 0$ then we are done and x_0 is optimal, otherwise we choose x_1 by the following:

Claim. *Suppose there is some $j \in N_0$ such that $\mu_{j,0} > 0$ and there is an $i \in B_0$ such that there is a basic feasible solution x_1 with basis $B_1 = B_0 \cup \{j\} \setminus \{i\}$. Then*

$$c^\top x_1 = c^\top x_0 + \mu_{j,0} x_{j,1} > c^\top x_0$$

Proof. If x is any feasible solution, then

$$c^\top x = c^\top x + \lambda_0^\top (b - Ax) = b^\top \lambda_0 + (c - A^\top \lambda_0)^\top x = c^\top x_0 + \sum_{k \in N_0} \mu_{k,0} x_k$$

as $\mu_{k,0} = 0$ for any $k \in B_0$. The sum is then

$$\sum_{k \in N_0} \mu_k x_{k,1} = \sum_{k \in N_1} \mu_k x_{k,1} + \mu_j x_{j,1} - \mu_i x_{i,1} = \mu_j x_{j,1}$$

Also $x_{j,1} > 0$ by non-degeneracy assumption. The claim follows. □

Proposition 3.4. *Suppose there is a $j \in N_0$ such that $\mu_{j,0} > 0$ and for every $i \in B_0$ the basis solution with basis $B_1 = B_0 \cup \{j\} \setminus \{i\}$ is infeasible, then the problem is unbounded.*

So if no feasible x_1 exists, then we can stop the algorithm and conclude that the problem is unbounded. Otherwise, we can repeat the algorithm again and again, and stop either at a basic feasible solution whose accompanied Lagrange multiplier is feasible in the dual problem or at a point where we can conclude the unboundedness of the problem. We may, however, have to go through all basic feasible solutions if we start with a bad guess, but at least we know after each step we shall obtain a better solution candidate.

3.6 Simplex Algorithm in Action

We are still considering the problem of maximising $c^\top x$ subject to $Ax = b, x \geq 0$, assuming the same things we assumed previously. Suppose we know one basic feasible solution x_0, we want to use the simplex algorithm. Before that, we need to do some pre-processing of the problem. Let B be the basis of x_0 and $N = \{1, \ldots, n\} \setminus B$ as usual. For $x \in \mathbb{R}^n$, we rearrange the components $x = (x_B, x_N)^\top$ as usual. Furthermore, the objective function can be written as

$$c^\top x = c^\top x_0 + \mu_{N,0}^\top x_N$$

for feasible x. where $\mu_0 = c - A^\top \lambda_0$ and $\lambda_0 = (A_B^\top)^{-1} c_B$ is the Lagrange multiplier obtained by complementary slackness. The set of feasible solutions is then subject to the condition

$$x_B + A_B^{-1} A_N x_N = x_{B,0}, x_B, x_N \geq 0$$

To use the simplex algorithm, we first formulate the initial simplex tableau as the $(m+1)\times(n+1)$ matrix (in practice some of the columns are shuffled due to the position of the basis)

$$\Gamma = \begin{pmatrix} I & A_B^{-1}A_N & x_{B,0} \\ 0 & \mu_{N,0}^\top & -c^\top x_0 \end{pmatrix}$$

where I is the $m \times m$ identity. We first test for optimality of x_0. If $\mu_0 \leq 0$, then we can stop as it would be optimal due to the fundamental theorem. Otherwise, we choose the pivot column, i.e. $j \in N$ such that $\mu_{j,0} > 0$. a rule of thumb is to pick j such that $\mu_{j,0}$ is the largest. We also pick the pivot row by looking within the pivot column j and find the $i \in B$ which minimises $x_{i,0}/\Gamma_{i,j}$ over all $i \in B$ with $\Gamma_{i,j} > 0$. If this cannot be done, i.e. $\Gamma_{i,j} \leq 0$ for all i, then we can stop as it indicates the unboundedness of the problem.
Assume we go through these steps without stopping the algorithm, we perform the pivot operation to move to the next basic feasible solution. To do this, we scale row i with $1/\Gamma_{i,j}$ and replace row k with

$$(\text{old row } k) - (\text{old row } i) \times \frac{\Gamma_{k,j}}{\Gamma_{i,j}}$$

for all $k \neq i$ and run the algorithm on this new simplex tableau.

Remark. 1. For the initial basic feasible solution we have $x_{i,0} > 0$ and $x_{j,0} = 0$, whilst for the next basic feasible solution $x_{i,1} = 0$ and $x_{j,1} = x_{i,0}/\Gamma_{i,j} > 0$.
2. The pivot operation is actually just a Gaussian elimination.
3. The Γ we obtained after the pivot operation is the same Γ had we start with x_1 instead.
4. Suppose $\Gamma_{i,j} \leq 0$ for all $i \in B$, then for $i \in B$ and for $r > 0$ let $x_r = x_0 + r(\delta_j - \Gamma_{i,j}\delta_i)$ where δ_k is the k^{th} standard basis. Then the transformation $x_0 \mapsto x_r$ replaces $x_{i,0}$ with $x_{i,r} = x_{i,0} - r\Gamma_{i,j}$ and $x_{j,0} = 0$ with $x_{j,r} = r$ and leaves other entries unchanged. Now x_r is feasible since $x_r \geq 0$ and

$$\begin{pmatrix} I & A_B^{-1}A_N \end{pmatrix} x_r = x_0$$

But $c^\top x_r = c^\top x_0 + r\mu_{j,0} \to \infty$ as $r \to \infty$, so the problem is unbounded. This proves Proposition 3.4.

Example 3.1. Consider the linear program to maximise $3x_1 + 2x_2$ subject to

$$\begin{cases} 2x_1 + x_2 \leq 4 \\ 2x_1 + 3x_2 \leq 6 \\ x_1, x_2 \geq 0 \end{cases}$$

Then its dual problem is to minimise $4\lambda_1 + 6\lambda_2$ subject to

$$\begin{cases} 2\lambda_1 + 2\lambda_2 \geq 3 \\ \lambda_1 + 3\lambda_2 \geq 2 \\ \lambda_1, \lambda_2 \geq 0 \end{cases}$$

By introducing slack variables, the primal problem has constraint

$$\begin{cases} 2x_1 + x_2 + z_1 = 4 \\ 2x_1 + 3x_2 + z_2 = 6 \\ x_1, x_2, z_1, z_2 \geq 0 \end{cases}$$

and the dual problem has constraint

$$\begin{cases} 2\lambda_1 + 2\lambda_2 - v_1 = 3 \\ \lambda_1 + 3\lambda_2 - v_2 = 2 \\ \lambda_1, \lambda_2, v_1, v_2 \geq 0 \end{cases}$$

So the complementary slackness translates to

$$x_1 v_1 = x_2 v_2 = \lambda_1 z_1 = \lambda_2 z_2 = 0$$

Running the simplex algorithm shall yield the maximiser $x_1 = 3/2, x_2 = 1$ with then corresponding dual solution $\lambda_1 = 5/4, \lambda_2 = 1/4$. We start with the simplex tableau (asterisked columns are where the identity matrix and the zero vectors are at)

			*	*	
	x_1	x_2	z_1	z_2	
z_1	2	1	1	0	4
z_2	2	3	0	1	6
payoff	3	2	0	0	0

Now the payoff row is not non-positive, hence we have not arrived at the optimal solution. As 3 is the largest payoff entry, we let x_1 enter the basis, so the first column is the pivot column. We choose the first row to be the pivot row. The the pivot operation yields the next simplex tableau:

	*			*	
	x_1	x_2	z_1	z_2	
x_1	1	1/2	1/2	0	2
z_2	0	2	-1	1	2
payoff	0	1/2	$-3/2$	0	-6

It is still not optimal due to the positive $1/2$ entry in the payoff row. So this time the pivot column is the second column and we choose the second row as the pivot row. This gives us

	*	*			
	x_1	x_2	z_1	z_2	
x_1	1	0	3/4	$-1/4$	3/2
x_2	0	1	$-1/2$	1/2	1
payoff	0	0	$-5/4$	$-1/4$	$-13/2$

which we realise is optimal as the payoff is nonpositive.

Remark. 1. If the linear program is a minimisation instead of a maximisation problem, then the stopping criterion would be the tableau being nonnegative.
2. When the problem is of the form like above, i.e. maximising $c^\top x$ subject to $Ax \leq b, x \geq 0$, one observe that the coefficient of the payoff row under the slack variables are minus the corresponding dual variables (via complementary slackness). For the final tableau, the feasible dual variables are the Lagrange multipliers for the problems.
3. In the example, we have

$$\begin{pmatrix} 3/4 & -1/4 \\ -1/2 & 1/2 \end{pmatrix} \begin{pmatrix} 2 & 1 & 1 & 0 & 4 \\ 2 & 3 & 0 & 1 & 6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 3/4 & -1/4 & 3/2 \\ 0 & 1 & -1/2 & 1/2 & 1 \end{pmatrix}$$

Hence the perturbed problem of maximising $3x_1 + 2x_2$ subject to

$$\begin{cases} 2x_1 + x_2 \leq 4 + \epsilon_1 \\ 2x_1 + 3x_2 \leq 6 + \epsilon_2 \\ x_1, x_2 \geq 0 \end{cases}$$

with $\epsilon_{1,2}$ small yields the final tableau

	$*$ x_1	$*$ x_2	z_1	z_2	
x_1	1	0	$3/4$	$-1/4$	$3/2 + 3\epsilon_1/4 - \epsilon_2/4$
x_2	0	1	$-1/2$	$1/2$	$1 - \epsilon_1/2 + \epsilon_2/2$
payoff	0	0	$-5/4$	$-1/4$	$-13/2 - 5\epsilon_1/4 - \epsilon_2/4$

Therefore it has maximiser at

$$x_1 = \frac{3}{2} + \frac{3}{4}\epsilon_1 - \frac{1}{4}\epsilon_1, x_2 = 1 - \frac{1}{2}\epsilon_1 + \frac{1}{2}\epsilon_2, 3x_1 + 2x_2 = \frac{13}{2} + \frac{5}{4}\epsilon_1 + \frac{1}{4}\epsilon_2$$

given that they are feasible (possible for small enough $\epsilon_{1,2}$).

3.7 The Two-Phase Algorithm

Again our problem is to maximise $c^\top x$ subject to $Ax = b, x \geq 0$. In order to roll out the simplex algorithm, we need at least one basic feasible solution. It is certainly tempting if we can know a way that can yield one efficiently.
There are a lot of approaches, for example just pick a basis and computed $A_B^{-1}b$ (using the notations we introduced earlier). This is certainly a sufficient algorithm, but this might not be very efficient.
Consider a new problem of minimising $e^\top y$ subject to $Ax + y = b, x, y \geq 0$ where $e = (1, \ldots, 1)^\top$. The variable y here is called an artificial variable. Now $(x, y) = (0, b)$ is a basic feasible solution for this problem, so we can apply the simplex algorithm which must terminate at some $(x, y) = (x_0, 0)$ given that the original problem is feasible. Then x_0 has to be basic feasible for the original problem. So we can modify the algorithm by adding a first phase to compute one basic feasible solution first (called Phase I) and then apply the simplex algorithm to it (which is Phase II).

Example 3.2. Consider the problem to maximise $x_1 - 3x_2 + 5x_3$ subject to

$$\begin{cases} x_1 + x_2 + x_3 \leq 30 \\ -x_2 + 2x_3 = 20 \\ -x_1 + 2x_2 + x_3 \geq 40 \\ x_1, x_2, x_3 \geq 0 \end{cases}$$

we introduce slack variables as usual to transform the constraint to

$$\begin{cases} x_1 + x_2 + x_3 + z_1 = 30 \\ -x_2 + 2x_3 = 20 \\ -x_1 + 2x_2 + x_3 - z_2 = 40 \\ x_1, x_2, x_3, z_1, z_2 \geq 0 \end{cases}$$

Add in artificial variable y_1, y_2,

$$\begin{cases} x_1 + x_2 + x_3 + z_1 = 30 \\ -x_2 + 2x_3 + y_1 = 20 \\ -x_1 + 2x_2 + x_3 - z_2 + y_2 = 40 \\ x_1, x_2, x_3, z_1, z_2, y_1, y_2 \geq 0 \end{cases}$$

In some sense we should have added a variable y_3 on the first expression as well, but it is actually redundant as we already have an nonnegative expression. We solve this new problem with the simplex algorithm, with the initial tableau:

				*		*	*	
	x_1	x_2	x_3	z_1	z_2	y_1	y_2	
z_1	1	1	1	1	0	0	0	30
y_1	0	-1	2	0	0	1	0	20
y_2	-1	2	1	0	-1	0	1	40
Phase II	1	-3	5	0	0	0	0	0
Phase I	0	0	0	0	0	1	1	0

We have put the objective functions of both phases so that we can manipulate both simultaneously so that when Phase I is completed, we can immediately go to Phase II.
Now the identity part of the upper part of the tableau does not correspond to the zero part of the Phase I payoff, so we do a transformation to modify the tableau:

				*		*	*	
	x_1	x_2	x_3	z_1	z_2	y_1	y_2	
z_1	1	1	1	1	0	0	0	30
y_1	0	-1	2	0	0	1	0	20
y_2	-1	2	1	0	-1	0	1	40
Phase II	1	-3	5	0	0	0	0	0
Phase I	1	-1	-3	0	1	0	0	-60

Note that as we are encountering a minimisation problem in Phase I, our aim is to make the last row nonnegative. Pivot once:

			*	*			*	
	x_1	x_2	x_3	z_1	z_2	y_1	y_2	
z_1	1	3/2	0	1	0	$-1/2$	0	20
x_3	0	$-1/2$	1	0	0	1/2	0	10
y_2	-1	5/2	0	0	-1	$-1/2$	1	30
Phase II	1	$-1/2$	0	0	0	$-5/2$	0	-50
Phase I	1	$-5/2$	0	0	1	3/2	0	-30

Pivot twice:

		*	*	*				
	x_1	x_2	x_3	z_1	z_2	y_1	y_2	
z_1	8/5	0	0	1	3/5	$-1/5$	$-3/5$	2
x_3	$-1/5$	0	1	0	$-1/5$	2/5	1/5	16
x_2	$-2/5$	1	0	0	$-2/5$	$-1/5$	2/5	12
Phase II	4/5	0	0	0	$-1/5$	$-13/5$	1/5	-44
Phase I	0	0	0	0	0	1	1	0

The last row is nonnegative and the objective function attains zero, so we have already achieved our aim for Phase I with the basic feasible solution shown in the tableau. This produces our initial tableau for Phase II:

		*	*	*		
	x_1	x_2	x_3	z_1	z_2	
z_1	8/5	0	0	1	3/5	2
x_3	−1/5	0	1	0	−1/5	16
x_2	−2/5	1	0	0	−2/5	12
Phase II	4/5	0	0	0	−1/5	−44

Now we are maximising, so we need to make the payoff nonpositive. After the first pivoting we have

	*	*	*			
	x_1	x_2	x_3	z_1	z_2	
x_1	1	0	0	5/8	3/8	5/4
x_3	0	0	1	1/8	−1/8	65/4
x_2	0	1	0	1/4	−1/4	25/2
Phase II	0	0	0	−1/2	−1/2	−45

which has nonpositive payoff, hence the optimal solution is at $(x_1, x_2, x_3) = (5/4, 25/2, 65/4)$.

4 Game Theory

4.1 Two-person Zero-sum Games

Consider a game where there are two players called Player I and Player II. Player I has m choices of strategies labeled by $i \in \{1, \dots, m\}$ and Player II has n choices of strategies $j \in \{1, \dots, n\}$. We assume that the game is zero-sum, i.e. if Player I chooses strategy i and Player II chooses strategy j, then Player I is paid $a_{i,j}$ and Player II is paid $-a_{i,j}$. So the net payment across the game is zero. The matrix $A = (a_{i,j})$ is called the payoff matrix of the game. This is an $m \times n$ matrix. Assume that both players want to maximise their payoff and they know the other wants it too. [2]
Player I might want to solve the maximisation of $\min_j a_{ij}$ subject to $i \in \{1, \dots, m\}$ and Player II to solve the minimisation of $\max_i a_{ij}$ subject to $j \in \{1, \dots, n\}$.

Example 4.1. Consider

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

Then obviously Player I will pick $i = 2$ and Player II will pick $j = 1$. This is independent of who playing first.

Such a point $(2, 1)$ in the above example with the said property is called a saddle point of the matrix.

[2] And they know the other knows they want it, and they know the other knows they know the other wants it, etc.

Definition 4.1. A saddle point of a payoff matrix A is a pair of strategies (i, j) with the property

$$a_{i,j} = \max_{i'} \min_{j'} a_{i',j'} = \min_{j'} \max_{i'} a_{i',j'}$$

If a payoff matrix A has saddle point (i, j), then $a_{i,j}$ is called the value of the game.

Not all payoff matrices have a saddle point, an easy example is

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

where $\max_i \min_j a_{i,j} = 2 \neq 3 = \min_j \max_i a_{i,j}$.

4.2 Pure and Mixed Strategies

Often, the game has the property that the players choose their strategies simultaneously. The idea to analyse these games is to randomise the strategies.

Definition 4.2. A mixed strategy is an assignment of probabilities to each of the individual strategies. A pure strategy is a mixed strategies assigning 1 to one particular strategy and 0 to all others.

We use the notation that Player I plays strategy i with probability p_i and Player II plays strategy j with probability q_j. Therefore Player I would want to maximise

$$\min_j \mathbb{E}[\text{payout}|\text{Player II picks strategy } j]$$

which translates to maximising

$$\min_j \sum_{i=1}^{m} p_i a_{i,j}$$

subject to

$$\sum_{i=1}^{m} p_i = 1, \forall i \in \{1, \dots, m\}, p_i \geq 0$$

We can turn this into a linear program. Introducing $e = (1, \dots, 1)^\top$ (dimension implied by context), then the problem can be written as maximising $v \in \mathbb{R}$ subject to $A^\top p \geq ve, e^\top p = 1, p \geq 0$. Similarly, the optimisation problem for Player II can be written as minimising w subject to $Aq \leq we, e^\top q = 1, q \geq 0$.
Unsurprisingly, these two problems are naturally dual to each other. Indeed, one can observe that the Lagrangian of Player I is

$$\begin{aligned} L(p, v, z, q, w) &= v + q^\top (A^\top p - z - ve) + w(1 - p^\top e) \\ &= w + v(1 - q^\top e) + p^\top (Aq - we) - z^\top q \end{aligned}$$

where z is the slack variable. So the feasible Lagrange multipliers are constrained by $q \geq 0, Aq \leq we, e^\top q = 1$ and $\sup L = w$.
By the fundamental theorem of linear programs, (p, v) is optimal iff there is some (q, w) such that both tuples are feasible and

$$(Ap - we)^\top p = 0 = q(A^\top p - ve)$$

Where we see $wp^\top e = vq^\top e$, so $v = w$ in optimal case as one would expect.

Theorem 4.1 (Fundamental Theorem of Matrix Games)**.** *The mixed row strategy p is optimal for Player I iff there exists a mixed column strategy q for Player II and $v \in \mathbb{R}$ such that:*
1. $A^\top p \geq ve, e^\top p = 1, p \geq 0$.
2. $Aq \leq ve, e^\top q = 1, q \geq 0$.
3. $v = p^\top Aq$. *In this case,* (p, v) *is optimal for the problem of Player I and* (q, v) *is optimal for Player II.*

Proof. Immediate. □

The value v above is called the value of the game. Hence, for optimal mxied strategies, complementary slackness ensures that if Player II plays j with positive probability, then the conditional expected payoff given Player II plays j equals the value of the game. An analogous statement works the other way around. In other words,

$$\begin{cases} q_j > 0 \implies (A^\top p)_j = v \\ p_i > 0 \implies (Aq)_i = v \end{cases}$$

Definition 4.3. A game is symmetric if the payoff matrix is antisymmetric. That is $A^\top = -A$.

Example 4.2. The rock-paper-scissors game

$$\begin{pmatrix} 0 & -1 & 1 \\ 1 & 0 & -1 \\ -1 & 1 & 0 \end{pmatrix}$$

is symmetric.

Theorem 4.2. *A symmetric game has zero value.*

This is very intuitive by symmetry. We can also make use of our oven-ready fundamental theorem.

Proof. Suppose (p, v, q) satisfy the (necessary) conditions of optimality, then by antisymmetry of A, the tuple $(q, -v, p)$ satisfies the (sufficient) condition of optimality. Hence $v = -v \implies v = 0$. □

4.3 Finding Optimal Strategies

If the game admits a saddle point (i, j), then by definition the pure strategy i is optimal for Player I and the strategy j is optimal for Player II, consequently the value of the game is $a_{i,j}$. Otherwise, we reduce the game by eliminating dominating strategies. A row i dominates i' is $a_{i,j} \geq a_{i',j}$ for any j, so if a row i' is dominated by some other row, then Player I will never play strategy i'. Similarly if there is some j, j' with $a_{i,j} \leq a_{i,j'}$ for any i, then Player II will never play j'.

Example 4.3. Consider the payoff matrix

$$A = \begin{pmatrix} 2 & 3 & 4 & 2 \\ 3 & 1 & 1/2 & 4 \\ 1 & 3 & 2 & 3 \end{pmatrix}$$

One can check that there is no saddle point. Also, the first column dominates the fourth column, so Player II will never play the fourth column. After eliminating that, the first row dominates the third row, so the third row is eliminated too, leaving a modified payoff matrix

$$A' = \begin{pmatrix} 2 & 3 & 4 \\ 3 & 1 & 1/2 \end{pmatrix}$$

So the optimal strategy p of Player I is of the form $(p_1, 1-p_1, 0)^\top$.
Now the region $(A')^\top p' \geq ve$ (where $p' = (p_1, 1-p_1)^\top$ are the remaining relevant strategies) is given by

$$\begin{cases} 2p_1 + 3(1-p_1) \geq v \implies v \leq 3 - p_1 \\ 3p_1 + (1-p_1) \geq v \implies v \leq 1 + 2p_1 \\ 4p_1 + (1-p_1)/2 \geq v \implies v \leq (1+7p_1)/2 \end{cases}$$

So by sketching the graph of v against p_1, we find that the maximum v occurs at $p_1 = 2/3$ and the optimal strategy of Player I is $p = (2/3, 1/3, 0)^\top$, and the value of the game is $v = 7/3$.
By complementary slackness $q_3 = 0$ as the third constraint is not tight. So $q = (q_1, 1-q_1, 0, 0)^\top$ for some probability q_1. As $p_1 = 2/3 > 0$, the first dual constraint is binding by complementary slackness, so

$$2q_1 + 3(1-q_1) = 7/3 \implies q_1 = 2/3$$

Therefore $q = (2/3, 1/3, 0, 0)^\top$.

If it does not get clear after using these tricks, one can always use the simplex algorithm. If $\min_{i,j} a_{i,j} > 0$ then we know v is strictly positive, hence we can rewrite the problem of Player I as maximising v subject to $A^\top x \geq e, e^\top x = 1/v, x \geq 0$ where $x = p/v$. This is equivalent to minimising $e^\top x$ subject to $A^\top x \geq e, x \geq 0$ which is a form where we can apply simplex algorithm to. Note that its dual problem is maximising $e^\top y$ subject to $Ay \leq e, y \geq 0$ which is in a form for Phase I of the two-phase algorithm.
If instead $\min_{i,j} a_{i,j} \leq 0$, we can still use the same idea. Just find some k such that $a'_{i,j} = a_{i,j} + k > 0$ for all i, j and roll out the same algorithm on $A' = (a'_{i,j})$. Then the optimal strategy of A, A' would be the same and the new value v' would satisfy $v = v' - k$ where v is the value of the original game.

Example 4.4. Consider the game of Rock-Paper-Scissors, i.e.

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

Then we consider the modified game

$$A' = \begin{pmatrix} 2 & 1 & 3 \\ 3 & 2 & 1 \\ 1 & 3 & 2 \end{pmatrix}$$

where $k = 2$. So the problem of Player II is to maximise $y_1 + y_2 + y_3$ subject to

$$\begin{cases} 2y_1 + y_2 + 3y_3 \leq 1 \\ 3y_1 + 2y_2 + y_3 \leq 1 \\ y_1 + 3y_2 + 2y_3 \leq 1 \\ y_1, y_2, y_3 \geq 0 \end{cases}$$

So the initial simplex tableau is

				*	*	*	
	y_1	y_2	y_3	z_1	z_2	z_3	
z_1	2	1	3	1	0	0	1
z_2	3	2	1	0	1	0	1
z_3	1	3	2	0	0	1	1
Payoff	1	1	1	0	0	0	0

and the final tableau would be

	*	*	*				
	y_1	y_2	y_3	z_1	z_2	z_3	
y_3	0	0	1	7/18	−5/18	1/18	1/6
y_1	1	0	0	1/18	7/18	−5/18	1/6
y_2	0	1	0	−5/18	1/18	7/18	1/6
Payoff	0	0	0	−1/6	−1/6	−1/6	−1/2

Therefore $v' = 2$, so $v = 0$.

5 Network Flow Problems

5.1 The Maximal Flow Problem

Definition 5.1. A network (or directed graph) is a set $N = \{1, \ldots, n\}$ of nodes (or vertices) together with a collection $A \subset N \times N$ of ordered pairs (i, j) called arcs or directed edges. The flow in the network is a collection of numbers $(x_{i,j})$ indexed by elements in $N \times N$.

The problem we wish to consider is formulated in the following way: Each arc (i, j) is assigned a capacity $c_{i,j} \geq 0$. In reality (or for notational convenience), we will pretend all arcs in $N \times N$ exists and $c_{i,j} = 0$ for $(i, j) \notin A$.
We distinguish two special nodes in the network. Node 1 is called the source node while the node n is called the sink. The maximum flow problem is to pump as much flow through the network, from the source to the sink, subject to the capacity constraints and the constraint that the flow-in and flow-out cancelled out in any notes except the source and the sink. [3] Formally, we want to maximise $v \in \mathbb{R}$ subject to $0 \leq x_{i,j} \leq c_{i,j}$ and

$$\sum_{j \in N} x_{i,j} - \sum_{j \in N} x_{j,i} = \begin{cases} v, \text{ if } i = 1 \\ 0, \text{ if } 1 < i < n \\ -v, \text{ if } i = n \end{cases}$$

[3] Normally, there is no flow-in at the source and no flow-out at the sink.

which, as one can discover joyfully, is a linear program. As an aside, the variable v does not have to appear as it can be expressed in terms of $(x_{i,j})$ in the constraint. It is simply for notational convenience.
Of course, we can apply the barrier method or simplex algorithm and call it a day, but certainly there are more useful structures in this system that allows us to simplify it.

5.2 The Ford-Fulkerson Algorithm

A nice algorithm developed by Ford-Fulkerson is as follows:
First, assign an initial feasible flow $(x_{i,j})$ with value v (for example we can take all of them to be zero). Then find an augmented path, defined a a collection of nodes $1 = i_1, i_2, \ldots, i_{r-1}, i_r = n$ with eiher $c_{i_k,i_{k+1}} - x_{i_k,i_{k+1}} > 0$ or $x_{i_{k+1},i_k} > 0$ for each k. If an augmented path exists, then increase the flow along the path by the amount

$$\delta = \min_{k\in\{1,\ldots,r-1\}} \max\{c_{i_k,i_{k+1}} - x_{i_k,i_{k+1}}, x_{i_{k+1},i_k}\}$$

which is positive by assumption and

$$x'_{i,j} = \begin{cases} x_{i,j} + \delta, \text{ if } i,j \text{ are consecutive in the path} \\ x_{i,j} - \delta, \text{ if } j,i \text{ are consecutive in the path} \\ x_{i,j}, \text{ otherwise} \end{cases}$$

Then, as one can check, $(x'_{i,j})$ is also feasible with $v' = v + \delta$. We shall show that the non-existence of an augmented path is equivalent to the maximality of the flow.

Definition 5.2. Let $S \subset N$ be a collection of nodes. The complement of S in N is denoted $\bar{S} = N \setminus S$. A cut seperating the source and the sink is the set of arcs $(i,j) \in S \times \bar{S}$ where $\{1,n\} \in S$. THe capacity of a cut $(S,\bar{S})$ is defined as

$$c(S,\bar{S}) = \sum_{i\in S, j\in\bar{S}} c_{i,j}$$

Theorem 5.1 (The max-flow min-cut Theorem)**.** *For a network, the maximum value of a feasible flow from the source to the sink equals to the minimum of all cut capacities of all cuts seperating the source and the sink.*

Proof. Fix any feasible flow $(x_{i,j})$. For any $S, T \subset N$ we write

$$f(S,T) = \sum_{i\in S, j\in T} x_{i,j}$$

be the total flow over arcs joining S and T. Now let $S, \bar{S}$ be a cut seperating

the source and the sink, the constraint tells us

$$\begin{aligned} v &= \sum_{i\in S, j\in N} x_{i,j} - \sum_{j\in N, i\in S} x_{ji} \\ &= f(S,N) - f(N,S) \\ &= (f(S,S) + f(S,\bar{S})) - (f(S,S) + f(\bar{S},S)) \\ &= f(S,\bar{S}) - f(\bar{S},S) \\ &\leq f(S,\bar{S}) \\ &\leq c(S,\bar{S}) \end{aligned}$$

Hence the value of any flow is less than or equal to the capacity of any cut. To show that they are actual equal, notice that the inequality above becomes an equality iff $x_{i,j} = c_{i,j}$ and $x_{j,i} = 0$ for all arcs (i,j) in $(S,\bar{S})$. SO let $(x^\star_{i,j})$ be a maximal flow (which exists by compactness). We define a set S by induction in the following criteria:
1. The source is in S.
2. If $i \in S$ and $x^\star_{i,j} < c_{i,j}$, then j in S.
3. If $i \in S$ and $x^\star_{j,i} > 0$, then $j \in S$.
Now the sink is in $\bar{S}$ since otherwise there would be a path (by the inductive process) from the source to the sink pumping strictly more flow, contradicting the maximality of $(x^\star_{i,j})$. The cut $(S,\bar{S})$ has the property we wanted. □

The same constructive idea shows that if no augmented path can be found, then the flow is maximal as one can construct such a cut which has capacity equal to the value of the flow.
Also, the algorithm is guaranteed to terminate if the capacities and elements of the initial flow are all integer (hence it also terminates if all are rational by multiplying an integer). This is because the δ we obtain from the algorithm is at least 1, so if the value of v increases indefinitely through the algorithm, it cannot be bounded, which contradicts the compactness of the domain of the problem. But if something is irrational, the algorithm may not terminate.

5.3 The Dual Problem

The Lagrangian of the maximal flow problem is (note that v can be expressed in terms of $(x_{i,j})$):

$$\begin{aligned} L &= v + \sum_i \lambda_i \left(\sum_j x_{i,j} - \sum_j x_{j,i} \right) - \lambda_1 v + \lambda_n v + \sum_{i,j} \mu_{i,j}(c_{i,j} - x_{i,j} - z_{i,j}) \\ &= \sum_{i,j} c_{i,j}\mu_{i,j} + v(1 - \lambda_1 + \lambda_n) + \sum_{i,j} x_{i,j}(\lambda_i - \lambda_j - \mu_{i,j}) - \sum_{i,j} \mu_{i,j} z_{i,j} \end{aligned}$$

where z is the slack variable. For L to be bounded above varying x, z, we have

$$\begin{cases} 1 - \lambda_1 + \lambda_n = 0 \\ \lambda_i - \lambda_j \leq \mu_{i,j} \\ \mu_{i,j} \geq 0 \end{cases}$$

So the dual problem is to minimise $\sum_{i,j} c_{i,j}\mu_{i,j}$ subject to $\lambda_1 - \lambda_n = 1, \lambda_i - \lambda_j \leq \mu_{i,j}, \mu_{i,j} \geq 0$. By complementary slackness, if x is optimal for the primal problem and (λ, μ) is optimal for the dual problem, then $\lambda_i - \lambda_j = \mu_{i,j}$ and $\mu_{i,j} = 0$.
By Theorem 5.1, there is a set S with $1 \in S, n \notin S$ such that the optimal dual variables are given by

$$\lambda_i = \begin{cases} 1, \text{ if } i \in S \\ 0, \text{ if } i \in \bar{S} \end{cases}, \mu_{i,j} = \begin{cases} 1, \text{ if } i \in S, j \in \bar{S} \\ 0, \text{ otherwise} \end{cases}$$

due to our previously discussed dual constraints.

5.4 Combinatorial Applications

There are certain combinatorial applications of Theorem 5.1. One combinatorial problems is formulated like this: Suppose that a standard deck of 52 playing cards dealt into 13 piles of 4 cards each, we want to show that it is possible to select exactly one card from each pile such that the selected cards contain one card of each of the 13 ranks A,2,...,10,J,Q,K.
To prove this, we build a network with nodes

$$A, a_1, \ldots, a_{13}, b_1, \ldots, b_{52}, c_1, \ldots, c_{13}, B$$

as follows: The nodes (a_i) represent the 13 piles, and the nodes (b_j) represent the 52 cards, and the node (c_k) represent the 13 ranks. The node A is the source and B is the sink.
We put an arc of capacity 1 from A to each of a_i, and an arc of infinite (or very large) capacity from a_i to b_j if card j is in stack i. There will be exactly four arcs leaving each a_i and exactly one arc entering each b_j.
We also put an arc of infinite capacity from b_j to c_k if card j has rank k, so again there will be exactly one arc leaving each of the nodes b_j and exactly four arcs entering each of the nodes c_k.
Lastly, put an arc of capacity 1 from each node c_k to the sink B.

Claim. *The minimum capacity of any cut is* 13.

Proof. The cut $\{(A, a_i) : i = 1, \ldots, 13\}$ has capacity 13. We shall show that any other cut has capacity at least 13. Consider a cut with finite capacity, then it does not contain any arc of the form (a_i, b_j) or (b_j, c_k). If there are r arcs of the form (c_k, B) for some k, then there are at least $s = 13 - r$ nodes c_k still connected to B. Each of these s many c_k's is connected to four of the nodes b_j exclusively, so $4s$ many b_j is connected to B. But then at least $(4s)/4 = s$ many nodes a_i are connected to those b_j, hence to B, and they are also connected to A. So there are at least s arcs of the form (A, a_i) to be removed in order to complete the cut. This produces a total capacity of at least $r + s = 13$. □

So by Theorem 5.1, there is a flow of value 13. But the capacities are either 1 or infinity (or a very big integer), the Ford-Fulkerson algorithm starting with initial flow 0 will terminate. Hence there is an optimal flow (whose flows along every arc must be 0 or 1 by assumption). Let x be the optimal flow, then necessarily $x_{A,a_i} = 1$. As it is a flow, there is some j such that $x_{a_i,b_j} = 1$ and some k such that $x_{b_j,c_k} = 1$. So we can match nodes in the way a_i, c_k which provides the solution.

5.5 The Transportation Problem

The problem is as follows: Imagine there are m suppliers of a good labelled $1, \ldots, m$. For any i, the supplier i can supply S_i units of the good. There are n destinations labelled $1, \ldots, n$ that require the supply of the good. For any j, destination j requires D_j units of the good. For each pair (i, j), there is an associated unit transport cost $d_{i,j}$. It is assumed in addition that the supply and demand matches, i.e.

$$\sum_{i=1}^{m} S_i = \sum_{j=1}^{n} D_j$$

This is basically the linear program to minimise the total cost $\sum_{i,j} d_{i,j} x_{i,j}$ subject to

$$\begin{cases} \forall i, \sum_{j=1}^{n} x_{i,j} = S_i \\ \forall j, \sum_{i=1}^{m} x_{i,j} = D_j \\ \forall i, j, x_{i,j} \geq 0 \end{cases}$$

The Lagrangian is then

$$\begin{aligned} L(x, \lambda, \mu) &= \sum_{i,j} d_{i,j} x_{i,j} + \sum_{i=1}^{m} \lambda_i \left(S_i - \sum_{j=1}^{n} x_{i,j} \right) + \sum_{j=1}^{n} \mu_j \left(D_j - \sum_{i=1}^{m} x_{i,j} \right) \\ &= \sum_{i,j} (d_{i,j} - \lambda_i - \mu_j) x_{i,j} + \sum_{i=1}^{m} \lambda_i S_i + \sum_{j=1}^{n} \mu_j D_j \end{aligned}$$

Note that $x_{i,j} \geq 0$, so we have $d_{i,j} \geq \lambda_i + \mu_j$ for the Lagrange multipliers to be feasible, hence

$$\Lambda = \{(\lambda, \mu) : \inf_{x \geq 0} L(x, \lambda, \mu) > -\infty\} = \{(\lambda, \mu) : \forall i, j, d_{i,j} \geq \lambda_i + \mu_j\}$$

If $(x_{i,j})$ is optimal for the problem, then there are feasible (λ, μ) such that

$$\forall i, j, (d_{i,j} - \lambda_i - \mu_j) x_{i,j} = 0$$

by complementary slackness. We can of course use simplex algorithm as it is a linear program, but the additional structure here allows us to simplify it.

5.6 The Transportation Algorithm

We design an algorithm for it as follows: First, find an initial feasible assignment. As there are $m + n - 1$ linearly independent relations constraining the feasible set, any basic feasible solution will have $m + n - 1$ basic variables. There are two ways to do this: the North-West method and the greedy algorithm. These will be introduced in a second.

After the initial feasible assignment is done, we assign Lagrange multipliers. We may take $\lambda_1 = 0$ and enforce complementary slackness by choosing λ_i and μ_i subject to $d_{i,j} = \lambda_i + \mu_j$ for each basic cell.

Next, we check for optimality by looking at the dual feasibility of the Lagrange multipliers. The feasibility constraint is just $d_{i,j} \geq \lambda_i + \mu_j$ for each i, j. If this is valid, then we are done.

Otherwise, we do the pivoting operation by picking one of the cells (i, j) such that $\lambda_i + \mu_j > d_{i,j}$. the rule is to pick the cell with the largest difference $\lambda_i + \mu_j - d_{i,j}$. Now put an amount $\epsilon > 0$ units of flow into the pivot cell. At the same time add or substract ϵ from the basic cell to maintain feasibility. We will try to choose the largest ϵ possible such that the flow is feasible. Then do the same thing again and again until we reach optimality.

Example 5.1. Suppose the table of supply, demand and cost is

$d_{i,j}$					D_j
	8	6	7	5	12
	4	3	5	4	8
	9	8	6	7	11
S_i	7	6	10	8	

The North-West method is filling in the number of unit in each transport flow starting from North-West (top-left) of the array and fit in as many as we can, and the precedence of filling is from top-left to bottom-right. This way would give the basic feasible solution:

$$\begin{pmatrix} 7 & 5 & & \\ & 1 & 7 & \\ & & 3 & 8 \end{pmatrix}$$

Then we assign Lagrange multipliers by complementary slackness:

	8	6	8	9	μ_j
0	7	5	0	0	
-3	0	1	7	0	
-2	0	0	3	8	
λ_i					

Now we test the optimality by writing out the payoffs, i.e. $\lambda_i + \mu_j - d_{i,j}$ in each of the zero cells (denoted by the terms in square brackets), which gives

	8	6	8	9	μ_j
0	7	5	[1]	[4]	
-3	[1]	1	7	[2]	
-2	[-3]	[-4]	3	8	
λ_i					

So we need to pivot the top right corner, i.e. make it basic. Suppose it is changed to ϵ, then we need to change our solution to

$$\begin{pmatrix} 7 & 5-\epsilon & & \epsilon \\ & 1+\epsilon & 7-\epsilon & \\ & & 3+\epsilon & 8-\epsilon \end{pmatrix}$$

which is always feasible. For the solution to be basic, we take $\epsilon = 5$ to swipe out an entry to get the new solution, which, after calculating Lagrange multipliers and payoffs,

	8	2	4	5	μ_j
0	7	[-4]	[-3]	5	
1	[5]	6	2	[2]	
2	[1]	[-4]	8	3	
λ_i					

So we pivot the second row of the first column with, as we can find, has $\epsilon = 2$. Doing this again and again and again and again and again and again and again and again and again yields our final table

	7	6	4	5	μ_j
0	$[-1]$	5	$[-3]$	7	
-3	7	1	$[-4]$	$[-2]$	
2	$[0]$	$[0]$	10	1	
λ_i					

where all the entries in the square bracket is nonpositive, hence this is the optimal solution.
The greedy algorithm works as well. This time we start with the cell of lowest cost and fill in as many as possible, then the second lowest, etc.. In our case, this will produce the initial feasible assignment

$$\begin{pmatrix} 6 & & & 6 \\ & 6 & & 2 \\ 1 & & 10 & \end{pmatrix}$$

to which we can apply the same method to get to optimality.

Remark. If the original posture of the prblem is such that the total supply exceeds the total demand, then we can reformulate the problem by adding another destination with demand

$$D_{n+1} = \sum_{i=1}^{m} S_i - \sum_{j=1}^{n} D_j$$

and set $d_{i,n+1} = 0$ for all i to ensure that all differences are consumed.

Why does pivoting decrease the total cost? For any feasible x and any collection (λ_i) and (μ_j) of Lagrange multipliers we have

$$\begin{aligned} f(x) &= L(x, \lambda, \mu) \\ &= \sum_{i,j} (d_{i,j} - \lambda_i - \mu_j) x_{i,j} + \sum_{i=1}^{m} \lambda_i S_i + \sum_{j=1}^{n} \mu_j D_j \end{aligned}$$

So suppose x_0 is the initial feasible assignment, then we can choose $\lambda_{i,0}$ and $\mu_{i,0}$ to enforce complementary slackness $(d_{i,j,0} - \lambda_{i,0} - \mu_{j,0}) x_{i,j,0} = 0$, then

$$f(x_0) = \sum_{i=1}^{m} \lambda_{i,0} S_i + \sum_{j=1}^{n} \mu_{j,0} D_j$$

Let $B = \{(i,j) : x_{i,j,0} > 0\}$ and $N = \{(i,j) : x_{i,j,0} = 0\}$ be the basis of the basic feasible solution x_0. We choose $\lambda_{i,0}$ and $\mu_{i,0}$ such that $\lambda_{i,0} + \mu_{j,0} = d_{i,j}$ for all $(i,j) \in B$. Suppose the next feasible assignment x_1 is non-degenerate, as one of the initial non-basic cells (the pivot cell) is made basic, there is a cell $(i_N, j_N) \in N$ such that

$$x_{i_N, j_N, 0} = 0, x_{i_N, j_N, 1} = \epsilon > 0$$

Then we obtain

$$\begin{aligned} f(x_1) &= \sum_{(i,j)\in B} (d_{i,j} - \lambda_{i,0} - \mu_{j,0}) x_{i,j,1} + \sum_{(i,j)\in N} (d_{i,j} - \lambda_{i,0} - \mu_{j,0}) x_{i,j,1} + f(x_0) \\ &= (d_{i_N,j_N} - \lambda_{i_N,0} - \mu_{j_N,0})\epsilon + f(x_0) \\ &> f(x_0) \end{aligned}$$

by assumption.