# Optimisation

Cambridge University Mathematical Tripos: Part IB

4th May 2024

## Contents


**1 Introduction and convex functions** **3**
- 1.1 Outline and definitions . . . 3
- 1.2 Convexity . . . 3
- 1.3 Unconstrained optimisation . . . 4
- 1.4 First-order conditions for convexity . . . 4
- 1.5 Second-order conditions for convexity . . . 5

**2 Optimisation algorithms** **6**
- 2.1 Gradient descent . . . 6
- 2.2 Smoothness assumption . . . 6
- 2.3 Strong convexity assumption . . . 8
- 2.4 Proving gradient descent . . . 10
- 2.5 Rate of convergence . . . 11
- 2.6 Condition numbers and oscillation . . . 11
- 2.7 Newton's method . . . 11
- 2.8 Barrier methods . . . 12

**3 Lagrange multipliers** **13**
- 3.1 Introduction and Lagrange sufficiency . . . 13
- 3.2 Using Lagrange multipliers in general . . . 14
- 3.3 Complementary slackness . . . 15
- 3.4 Weak duality . . . 17
- 3.5 Strong duality and the Lagrange method . . . 17
- 3.6 Hyperplane condition for strong duality . . . 18
- 3.7 Strong duality and convex functions . . . 19
- 3.8 Shadow prices interpretation of Lagrange multipliers . . . 19

**4 Linear programming** **21**
- 4.1 Linear programs . . . 21
- 4.2 Maximising convex functions . . . 22
- 4.3 Basic solutions and basic feasible solutions . . . 23
- 4.4 Extreme points of the feasible set in standard form . . . 24

**5 Duality in linear programming** **24**
- 5.1 Strong duality of linear programs . . . 24
- 5.2 Duals of linear programs in standard form . . . 25

5.3 Duals of linear programs in general form . . . 25
5.4 Dual of dual program . . . 26
5.5 Dual of arbitrary linear program . . . 27
5.6 Optimality conditions . . . 27

**6 Simplex method** **28**
6.1 Introduction . . . 28
6.2 Feasibility of basic directions . . . 29
6.3 Cost of basic directions . . . 29
6.4 Moving to basic feasible solutions . . . 30
6.5 Simplex method . . . 31
6.6 Tableau implementation . . . 31

**7 Game theory** **34**
7.1 Zero-sum games . . . 34
7.2 Mixed strategies . . . 35
7.3 Duality of mixed strategy problems . . . 36
7.4 Finding optimal strategies . . . 36

**8 Network flows** **38**
8.1 Minimum cost flow . . . 38
8.2 Transport problem . . . 39
8.3 Sufficiency of transport problem . . . 39
8.4 Optimality conditions for transport problem . . . 40

**9 The transport algorithm** **40**
9.1 Transportation tableaux . . . 40
9.2 Updating the transportation tableau . . . 42

**10 Maximum flow, minimum cut** **44**
10.1 Introduction . . . 44
10.2 Cuts and flows . . . 44
10.3 Max-flow min-cut theorem . . . 45
10.4 Ford–Fulkerson algorithm . . . 45
10.5 Termination of Ford–Fulkerson . . . 48
10.6 Bipartite matching problem . . . 48

# 1 Introduction and convex functions

## 1.1 Outline and definitions

An *optimisation problem* is a problem in which we want to minimise some function $f(\mathbf{x})$ such that $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$. We may have a set of *constraints* $h(\mathbf{x}) = \mathbf{b}$ where $h(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^m$. Note that we will only ever consider minimisation of functions since we can maximise a function by minimising its negative. Such a problem is often written with notation such as

$$\begin{aligned} \underset{\mathbf{x}\in\mathcal{X}}{\text{minimise}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & h(\mathbf{x}) = \mathbf{b} \end{aligned}$$

**Definition.** The following definitions will be used.

(i) The function $f$ that we want to minimise is called the *objective function*.
(ii) The components of the vector $\mathbf{x}$ are called the *decision variables*.
(iii) A constraint of the form $h(\mathbf{x}) = \mathbf{b}$ is called a *functional constraint*.
(iv) A constraint of the form $\mathbf{x} \in \mathcal{X}$ is called a *regional constraint*.
(v) The set $\mathcal{X}(\mathbf{b}) = \{\mathbf{x} : \mathbf{x} \in \mathcal{X}, h(\mathbf{x}) = \mathbf{b}\}$ is called the *feasible set*.
(vi) If the feasible set is non-empty, the optimisation problem is called feasible. If the feasible set is empty, the problem is infeasible.
(vii) The problem is called *bounded* if the minimum on $\mathcal{X}(\mathbf{b})$ is bounded.
(viii) A point $\mathbf{x}^\star \in \mathcal{X}(\mathbf{b})$ is *optimal* if it minimises $f$ over $\mathcal{X}(\mathbf{b})$. The value $f(\mathbf{x}^\star)$ is called the optimal cost.

We can convert an inequality constraint into an equality constraint with a regional constraint, for instance

$$h(\mathbf{x}) \leq b \longrightarrow h(\mathbf{x}) + s = b;\ s \geq 0$$

## 1.2 Convexity

**Definition.** A set $S \subseteq \mathbb{R}^n$ is *convex* if for all $\mathbf{x}, \mathbf{y} \in S$, the line segment from $\mathbf{x}$ to $\mathbf{y}$ lies entirely inside $S$. In other words, for all $\lambda \in [0, 1]$, $\mathbf{x}(1 - \lambda) + \mathbf{y}(\lambda) \in S$.

**Definition.** A function $f : S \to \mathbb{R}$ is convex if

- $S$ is convex, and
- for all $\mathbf{x}, \mathbf{y} \in S$,

$$f((1 - \lambda)\mathbf{x} + \lambda\mathbf{y}) \leq (1 - \lambda)f(\mathbf{x}) + \lambda f(\mathbf{y})$$

So informally, for a convex function, if we take two inputs, the chord connecting their outputs lies above the function's curve. If the given inequality above is strict, the function is called *strictly* convex. $f$ is (strictly) *concave* if $-f$ is (strictly) convex. Note that if $f$ is linear, $f$ is convex and concave, since $f$ is linear in its input. Hence linear optimisation is a special case of convex optimisation.

## 1.3 Unconstrained optimisation

The *unconstrained* optimisation problem is simply to minimise $f(\mathbf{x})$, where $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function. Convex functions allow you to generalise the behaviour of a function in a small neighbourhood to global behaviour, so it becomes easier to solve optimisation problems expressed in terms of convex functions.

## 1.4 First-order conditions for convexity

Suppose we have a tangent to a curve $f : \mathbb{R} \to \mathbb{R}$ at a given point $x$. If $f$ is convex, then $f$ must only touch the curve once, since if it touched twice we would contradict the definition of convexity. In particular, we have the following necessary and sufficient condition for convexity:

$$f(y) \geq f(x) + (y - x)f'(x)$$

In higher dimensions, we might guess that

$$f(\mathbf{y}) \geq f(\mathbf{x}) + (\mathbf{y} - \mathbf{x}) \cdot \nabla f(\mathbf{x})$$

**Theorem.** A differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if and only if

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n,\ f(\mathbf{y}) \geq f(\mathbf{x}) + (\mathbf{y} - \mathbf{x}) \cdot \nabla f(\mathbf{x}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal (\mathbf{y} - \mathbf{x})$$

*Remark.* If $\nabla f(\mathbf{x}) = \mathbf{0}$ for some vector $\mathbf{x}$, then the first-order condition implies that $f(\mathbf{y}) \geq f(\mathbf{x})$, so $\mathbf{x}$ is the global minimum of $f$. This is an example of how we can use local properties (the gradient of the function at $\mathbf{x}$) to deduce global properties (the minimum value of the function).

*Proof.* First, we will prove that convexity implies the first-order condition. By convexity, we have

$$f((1 - \lambda)\mathbf{x} + \lambda\mathbf{y}) \leq (1 - \lambda)f(\mathbf{x}) + \lambda f(\mathbf{y})$$

Initially, let $n = 1$ so that we have the one-dimensional case. We have

$$f(y) \geq f(x) + \frac{f(x + \lambda(y - x)) - f(x)}{\lambda} = f(x) + \frac{f(x + \lambda(y - x)) - f(x)}{\lambda(y - x)}(y - x)$$

Hence, taking the limit as $\lambda \to 0$, we have

$$f(y) \geq f(x) + f'(x)(y - x)$$

For the general case, we define a function $g$ such that $g(\lambda) = f((1 - \lambda)\mathbf{x} + \lambda\mathbf{y})$. Since $f$ is convex, so is $g$. We can calculate

$$g'(\lambda) = \nabla f((1 - \lambda)\mathbf{x} + \lambda\mathbf{y}) \cdot (\mathbf{y} - \mathbf{x})$$

Since $g : [0, 1] \to \mathbb{R}$ is convex, by the above argument for $n = 1$ we have

$$g(1) \geq g(0) + g'(0)(1 - 0)$$
$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x}) \cdot (\mathbf{y} - \mathbf{x})$$

Now we must prove the converse; if the first-order condition holds, then $f$ is convex. Let

$$\mathbf{x}_\lambda = (1 - \lambda)\mathbf{x} + \lambda\mathbf{y}$$

The first-order condition shows that

$$f(\mathbf{x}) \geq f(\mathbf{x}_\lambda) + \nabla f(\mathbf{x}_\lambda) \cdot (\mathbf{x} - \mathbf{x}_\lambda)$$
$$f(\mathbf{y}) \geq f(\mathbf{x}_\lambda) + \nabla f(\mathbf{x}_\lambda) \cdot (\mathbf{y} - \mathbf{x}_\lambda)$$

Multiplying the first equation by $1 - \lambda$ and multiplying the second equation by $\lambda$, we get

$$\begin{aligned}
(1-\lambda)f(\mathbf{x}) + \lambda f(\mathbf{y}) &\geq f(\mathbf{x}_\lambda) + \nabla f(\mathbf{x}_\lambda) \cdot [(\mathbf{x} - \mathbf{x}_\lambda)(1-\lambda) + (\mathbf{y} - \mathbf{x}_\lambda)\lambda] \\
&= f(\mathbf{x}_\lambda) + \nabla f(\mathbf{x}_\lambda) \cdot [(\mathbf{x} - (1-\lambda)\mathbf{x} - \lambda\mathbf{y})(1-\lambda) + (\mathbf{y} - (1-\lambda)\mathbf{x} - \lambda\mathbf{y})\lambda] \\
&= f(\mathbf{x}_\lambda) + \nabla f(\mathbf{x}_\lambda) \cdot [(\lambda\mathbf{x} - \lambda\mathbf{y})(1-\lambda) + ((1-\lambda)\mathbf{y} - (1-\lambda)\mathbf{x})\lambda] \\
&= f(\mathbf{x}_\lambda) + \nabla f(\mathbf{x}_\lambda) \cdot 0 \\
&= f(\mathbf{x}_\lambda)
\end{aligned}$$

Hence $f$ really is convex. □

## 1.5 Second-order conditions for convexity

When $n = 1$, we suspect that $f''(x) \geq 0$ is the condition for convexity. In higher dimensions, the analogous operator to the double derivative is the Hessian matrix.

$$\nabla^2 f(\mathbf{x}) = \mathbf{H}_f = \begin{pmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1^2} & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_2 \partial x_1} & \frac{\partial^2 f(\mathbf{x})}{\partial x_2^2} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_1} & \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_n^2} \end{pmatrix}$$

**Definition.** An $n \times n$ matrix $A$ is *positive semidefinite* if for all $\mathbf{x} \in \mathbb{R}^n$, we have $\mathbf{x}^\intercal A \mathbf{x} \geq 0$. Equivalently, all eigenvalues of $A$ are non-negative. If $A$ is positive semidefinite, we write $A \succeq 0$.

Note that the higher-dimensional analogue of the Taylor expansion of $f(\mathbf{y})$ is

$$f(\mathbf{y}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal (\mathbf{y} - \mathbf{x}) + \frac{1}{2}(\mathbf{y} - \mathbf{x})^\intercal \nabla^2 f(\mathbf{x})(\mathbf{y} - \mathbf{x}) + \cdots$$

**Theorem.** A twice-differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if $\nabla^2 f(\mathbf{x}) \succeq 0$ at all $\mathbf{x}$. The converse also holds, but it is not important for this course, so it will not be proven.

*Proof.* Using the Taylor expansion of $f$, we have

$$f(\mathbf{y}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal (\mathbf{y} - \mathbf{x}) + \frac{1}{2}(\mathbf{y} - \mathbf{x})^\intercal \nabla^2 f(\mathbf{z})(\mathbf{y} - \mathbf{x})$$

where $\mathbf{z} = (1-\lambda)\mathbf{x} + \lambda\mathbf{y}$ for some $\lambda \in [0, 1]$. The rightmost term is positive, hence

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal (\mathbf{y} - \mathbf{x})$$

So the first-order conditions are satisfied, which imply convexity. □

# 2 Optimisation algorithms

## 2.1 Gradient descent

Consider minimising $f(x)$ such that $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function. Recall that a local minimum of $f$ is also the global minimum. Consider the following 'greedy' method:

- Start at a point $\mathbf{x}_0$.
- Search for close points around $\mathbf{x}_0$ whose values of $f$ are smaller than $f(\mathbf{x}_0)$.
  - If such a point exists, let this be $\mathbf{x}_1$. Repeat the algorithm.
  - If such a point does not exist, we have found a local minimum, which is the global minimum.

We can find such $\mathbf{x}_1$ points by considering the Taylor series expansion of $f$ around a point.

$$f(\mathbf{x} - \varepsilon\nabla f(\mathbf{x})) \approx f(\mathbf{x}) - \varepsilon\nabla f(\mathbf{x})^\intercal \cdot \nabla f(\mathbf{x}) = f(\mathbf{x}) - \varepsilon\|\nabla f(\mathbf{x})\|^2 \leq f(\mathbf{x})$$

Hence $-\nabla f(\mathbf{x})$ is called a descending direction. Although the gradient of the function is the most natural way of decreasing a function, any $\mathbf{v}$ with $f(\mathbf{x}) \cdot \mathbf{v} < 0$ is a descending direction. This gives us the *gradient descent* algorithm.

**Algorithm 1:** Gradient Descent Algorithm

**Result:** Global minimum of $f(\mathbf{x})$
start at a point $\mathbf{x}_0$;
$t \leftarrow 0$;
**repeat**
  find a descending direction $\mathbf{v}_t$, e.g. $-\nabla f(\mathbf{x})$;
  choose a step size $\eta_t$;
  $\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t + \eta_t \mathbf{v}_t$;
**until** $\nabla f(\mathbf{x}) = 0$ *or t is large enough*;

Different choices of $\mathbf{v}_t$ and $\eta_t$ give rise to many different qualities of algorithm.

## 2.2 Smoothness assumption

Some restrictions must be applied to a function to let us prove that gradient descent works.

**Definition.** A continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is *$\beta$-smooth* if $\nabla f$ is a $\beta$-Lipschitz function:

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq \beta\|\mathbf{x} - \mathbf{y}\|$$

In the following sections, we assume all functions $f$ are $\beta$-smooth. Further, if $f$ is twice differentiable (i.e. the Hessian exists everywhere), then the $\beta$-smoothness assumption is equivalent to

$$\nabla^2 f(\mathbf{x}) \preceq \beta I$$

so all eigenvalues of $\nabla^2 f(\mathbf{x})$ have $\lambda \leq \beta$. Also,

$$\mathbf{u}^\intercal \nabla^2 f(\mathbf{x})\mathbf{u} \leq \mathbf{u}^\intercal(\beta I)\mathbf{u} = \beta\|\mathbf{u}\|^2$$

**Definition.** The *linear approximation* to $f$ at $\mathbf{x}$ is

$$f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal (\mathbf{y} - \mathbf{x})$$

We might assume that the linear approximation is close to the actual function in a small neighbourhood around $x$.

**Claim.** If $f$ is $\beta$-smooth, then

$$f(\mathbf{y}) \leq f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal (\mathbf{y} - \mathbf{x}) + \frac{\beta}{2}\|\mathbf{y} - \mathbf{x}\|^2$$

Note that

$$f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal (\mathbf{y} - \mathbf{x}) \leq f(\mathbf{y})$$

since $f$ is convex, so this claim would show that $f$ really is close to the actual function, deviating by an arbitrarily small amount as we let $\mathbf{x}$ approach $\mathbf{y}$.

*Proof.* By Taylor's theorem,

$$\begin{aligned} f(\mathbf{y}) &= f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal (\mathbf{y} - \mathbf{x}) + \frac{1}{2}(\mathbf{y} - \mathbf{x})^\intercal \nabla^2 f(\mathbf{z})(\mathbf{y} - \mathbf{x}) \\ &\leq f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal (\mathbf{y} - \mathbf{x}) + \frac{1}{2}(\mathbf{y} - \mathbf{x})^\intercal (\beta I)(\mathbf{y} - \mathbf{x}) \\ &= f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal (\mathbf{y} - \mathbf{x}) + \frac{\beta}{2}\|\mathbf{y} - \mathbf{x}\|^2 \end{aligned}$$

□

**Corollary.** If we move by a step size of $\frac{1}{\beta}$, we will descend by at least $\frac{1}{2\beta}\|\nabla f(x)\|^2$.

$$f\left(\mathbf{x} - \frac{1}{\beta}\nabla f(\mathbf{x})\right) \leq f(\mathbf{x}) - \frac{1}{2\beta}\|\nabla f(x)\|^2$$

*Proof.* Consider

$$f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal (\mathbf{y} - \mathbf{x}) + \frac{\beta}{2}\|\mathbf{y} - \mathbf{x}\|^2$$

as a function of $\mathbf{y}$, and try to minimise it for a fixed $\mathbf{x}$.

$$\nabla_{\mathbf{y}}\left(f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal (\mathbf{y} - \mathbf{x}) + \frac{\beta}{2}\|\mathbf{y} - \mathbf{x}\|^2\right) = \nabla f(\mathbf{x}) + \beta(\mathbf{y} - \mathbf{x}) = 0$$

Hence,

$$\begin{aligned} \frac{\nabla f(\mathbf{x})}{\beta} &= \mathbf{x} - \mathbf{y} \\ \mathbf{y} &= \mathbf{x} - \frac{1}{\beta}\nabla f(\mathbf{x}) \end{aligned}$$

Substituting into the claim above, we have

$$\begin{aligned} f\Big(x - \frac{1}{\beta}\nabla f(\mathbf{x})\Big) &\leq f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal\Big(\frac{-1}{\beta}\nabla f(\mathbf{x})\Big) + \frac{\beta}{2}\Big\|\frac{1}{\beta}\nabla f(\mathbf{x})\Big\|^2 \\ &= f(\mathbf{x}) - \frac{1}{\beta}\|\nabla f(\mathbf{x})\|^2 + \frac{1}{2\beta}\|\nabla f(\mathbf{x})\|^2 \\ &= f(\mathbf{x}) - \frac{1}{2\beta}\|\nabla f(\mathbf{x})\|^2 \end{aligned}$$

□

**Claim** (Improved first order condition)**.**

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal(\mathbf{y} - \mathbf{x}) + \frac{1}{2\beta}\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|^2$$

*Proof.* For any $\mathbf{z}$, by the standard first order condition and the corollary above we have

$$f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal(\mathbf{z} - \mathbf{x}) \leq f(\mathbf{z}) \leq f(\mathbf{y}) + \nabla f(\mathbf{y})^\intercal(\mathbf{z} - \mathbf{y}) + \frac{\beta}{2}\|\mathbf{z} - \mathbf{y}\|^2$$

This then implies

$$f(\mathbf{x}) - f(\mathbf{y}) \leq \nabla f(\mathbf{x})^\intercal(\mathbf{x} - \mathbf{z}) + \nabla f(\mathbf{y})^\intercal(\mathbf{z} - \mathbf{y}) + \frac{\beta}{2}\|\mathbf{z} - \mathbf{y}\|^2$$

The left hand side is not dependent on $\mathbf{z}$, so by minimising $\mathbf{z}$ we get the best bound for the left hand side. We set the gradient of $\mathbf{z}$ to zero.

$$\begin{aligned} &-\nabla f(\mathbf{x}) + \nabla f(y) + \beta(\mathbf{z} - \mathbf{y}) = 0 \\ \implies &\mathbf{z} = \frac{\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})}{\beta} + \mathbf{y} \end{aligned}$$

Substituting back, we have

$$f(\mathbf{x}) - f(\mathbf{y}) \leq \nabla f(\mathbf{x})^\intercal(\mathbf{x} - \mathbf{y}) - \frac{1}{2\beta}\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|^2$$

□

## 2.3 Strong convexity assumption

In general, a small gradient does not imply that we are close to the optimum value of the function. We must therefore add an additional assumption in order to justify gradient descent.

**Definition.** A function $f : \mathbb{R}^n \to \mathbb{R}$ is called *$\alpha$-strongly convex* if

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal(\mathbf{y} - \mathbf{x}) + \frac{\alpha}{2}\|\mathbf{y} - \mathbf{x}\|^2$$

If $f$ is twice differentiable, then its Hessian satisfies

$$\nabla^2 f(\mathbf{x}) \succeq \alpha I$$

for all $\mathbf{x}$.

**Claim.** Let $f$ be $\alpha$-strongly convex. Let $p^\star$ be the optimal cost; i.e. the minimum value of $f$. Then for any $\mathbf{x}$ we have

$$p^\star \geq f(\mathbf{x}) - \frac{1}{2\alpha}\|\nabla f(\mathbf{x})\|^2$$

*Remark.* If $\|\nabla f(\mathbf{x})\| \leq \sqrt{2\alpha\varepsilon}$, then

$$p^\star \leq f(\mathbf{x}) \leq p^\star + \varepsilon$$

So a small gradient means we are close to the optimum.

*Proof.* The $\alpha$-strong convexity assumption gives

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal(\mathbf{y} - \mathbf{x}) + \frac{\alpha}{2}\|\mathbf{y} - \mathbf{x}\|^2$$

Taking the minimum over $\mathbf{y}$ of both sides, the left hand side becomes $p^\star$. Setting the gradient of the right hand side to zero,

$$\begin{aligned}\nabla f(\mathbf{x}) - \alpha(\mathbf{x} - \mathbf{y}) &= 0 \\ \frac{\nabla f(\mathbf{x})}{\alpha} &= (\mathbf{x} - \mathbf{y})\end{aligned}$$

This gives

$$\begin{aligned}p^\star &\geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal\left(-\frac{\nabla f(\mathbf{x})}{\alpha}\right) + \frac{\alpha}{2}\left\|\frac{\nabla f(\mathbf{x})}{\alpha}\right\|^2 \\ &= f(\mathbf{x}) - \frac{\|\nabla f(\mathbf{x})\|^2}{\alpha} + \frac{\|\nabla f(\mathbf{x})\|^2}{2\alpha} \\ &= f(\mathbf{x}) - \frac{\|\nabla f(\mathbf{x})\|^2}{2\alpha}\end{aligned}$$

□

**Claim.** Let $\mathbf{x}^\star$ be the minimising value, i.e. $f(\mathbf{x}^\star) = p^\star$. Then

$$\|\mathbf{x} - \mathbf{x}^\star\| \leq \frac{2}{\alpha}\|\nabla(\mathbf{x})\|$$

So if a function is strongly convex, we can find a region in which we know the global maximum lies.

*Proof.* By the Cauchy–Schwarz inequality,

$$\begin{aligned} f(\mathbf{x}^\star) &\geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\intercal (\mathbf{x}^\star - \mathbf{x}) + \frac{\alpha}{2}\|\mathbf{x}^\star - \mathbf{x}\|^2 \\ &\geq f(\mathbf{x}) - \|\nabla f(\mathbf{x})\|\|\mathbf{x}^\star - \mathbf{x}\| + \frac{\alpha}{2}\|\mathbf{x}^\star - \mathbf{x}\|^2 \end{aligned}$$

Since $f(\mathbf{x}^\star) \leq f(\mathbf{x})$, we have

$$0 \geq f(\mathbf{x}^\star) - f(\mathbf{x}) \geq -\|\nabla f(\mathbf{x})\|\|\mathbf{x}^\star - \mathbf{x}\| + \frac{\alpha}{2}\|\mathbf{x}^\star - \mathbf{x}\|^2$$

Hence,

$$\begin{aligned} \|\nabla f(\mathbf{x})\|\|\mathbf{x}^\star - \mathbf{x}\| &\geq \frac{\alpha}{2}\|\mathbf{x}^\star - \mathbf{x}\|^2 \\ \|\nabla f(\mathbf{x})\| &\geq \frac{\alpha}{2}\|\mathbf{x}^\star - \mathbf{x}\| \end{aligned}$$

□

## 2.4 Proving gradient descent

Let $f$ be a $\beta$-smooth and $\alpha$-strongly convex, where $0 < \alpha < \beta$. Then

$$\alpha I \preceq \nabla^2 f(\mathbf{x}) \preceq \beta I$$

**Theorem.** Gradient descent with step size $\frac{1}{\beta}$ satisfies

$$\begin{aligned} f(\mathbf{x}_T) - f(\mathbf{x}^\star) &\leq \left(1 - \frac{\alpha}{\beta}\right)^T (f(\mathbf{x}_0) - f(\mathbf{x}^\star)) \\ &\leq e^{-\frac{\alpha T}{\beta}} (f(\mathbf{x}_0) - f(\mathbf{x}^\star)) \\ &\leq e^{-\frac{\alpha T}{\beta}} \frac{\beta}{2}\|\mathbf{x}^\star - \mathbf{x}_0\|^2 \end{aligned}$$

*Proof.*

$$\begin{aligned} f(\mathbf{x}_{t+1}) - f(\mathbf{x}^\star) &\leq f(\mathbf{x}_t) - f(\mathbf{x}^\star) - \frac{1}{2\beta}\|\nabla f(\mathbf{x}_t)\|^2 \\ &\leq f(\mathbf{x}_t) - f(\mathbf{x}^\star) - \frac{\alpha}{\beta}(f(\mathbf{x}_t) - f(\mathbf{x}^\star)) \\ &\leq \left(1 - \frac{\alpha}{\beta}\right)(f(\mathbf{x}_t) - f(\mathbf{x}^\star)) \end{aligned}$$

Hence by induction,

$$f(\mathbf{x}_T) - f(\mathbf{x}^\star) \leq \left(1 - \frac{\alpha}{\beta}\right)^T (f(\mathbf{x}_0) - f(\mathbf{x}^\star))$$

The second line of the theorem is a consequence of the properties of the exponential function. The last inequality in the theorem can be shown by $\beta$-smoothness.

$$f(\mathbf{x}_0) \leq f(\mathbf{x}^\star) + \nabla f(\mathbf{x}^\star)^\intercal(\mathbf{x}_0 - \mathbf{x}^\star) + \frac{\beta}{2}\|\mathbf{x}_0 - \mathbf{x}^\star\|^2$$
$$f(\mathbf{x}_0) - f(\mathbf{x}^\star) \leq \frac{\beta}{2}\|\mathbf{x}_0 - \mathbf{x}^\star\|^2$$

□

## 2.5 Rate of convergence

For example, suppose that we would like $f(\mathbf{x}_T) - f(\mathbf{x}^\star) \leq 0.1$, and it takes $k$ steps to reach this tolerance. Then, it would take around $2k$ steps to reach a tolerance of 0.01, since the $\left(1 - \frac{\alpha}{\beta}\right)^T$ power might increase by a factor of 2. In general, the number of steps needed to ensure that the error is less than $\varepsilon$ is

$$T = \frac{\beta}{\alpha}\log\left(\frac{f(\mathbf{x}_0) - f(\mathbf{x}^\star)}{\varepsilon}\right)$$

This $\log(1/\varepsilon)$ term is called 'linear convergence', since for each extra order of magnitude of accuracy, we need a linear amount of computation steps. Linear convergence is very fast, and such algorithms are very useful.

## 2.6 Condition numbers and oscillation

Note that

$$1 - \frac{\alpha}{\beta}$$

is the term which controls the convergence of gradient descent. We call $\beta/\alpha$ the *condition number* of $f$. Such a number is always greater than 1. If the condition number is very close to 1, the convergence is fast. Consider the function

$$f(x_1, x_2) = \frac{1}{2}(x_1^2 + 100x_2^2)$$

The Hessian of $f$ at any point is

$$\nabla^2 f(x_1, x_2) = \begin{pmatrix} 1 & 0 \\ 0 & 100 \end{pmatrix}$$

Hence, $\alpha = 1, \beta = 100$ giving a condition number of 100. This function would optimise very slowly, and we may continually overshoot in the $x_2$ direction since the gradient points so strongly in this direction. We may like to prevent this oscillation between over-guessing and under-guessing certain coordinate components.

## 2.7 Newton's method

In gradient descent, we have

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta_t \nabla f(\mathbf{x}_t)$$

In Newton's method, we replace this formula with

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \left(\nabla^2 f(\mathbf{x})\right)^{-1} \nabla f(\mathbf{x}_t)$$

Note that the second order approximation for $f$ is

$$f(\mathbf{x}) \approx f(\mathbf{x}_t) + \nabla f(\mathbf{x}_t)^\intercal + \frac{1}{2}(\mathbf{x} - \mathbf{x}_t)^\intercal \nabla^2 f(\mathbf{x}_t)(\mathbf{x} - \mathbf{x}_t)$$

So if we instead try to minimise the right hand side of the second-order approximation with respect to $\mathbf{x}$, we have

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \left(\nabla^2 f(\mathbf{x})\right)^{-1} \nabla f(\mathbf{x}_t)$$

as given by Newton's method. This ideally allows us to deal with 'badly-proportioned' coordinates independently, by scaling each coordinate using the Hessian rather than by a constant. Essentially, Newton's method iteratively approximates the function with a parabola, and then moves to the minimum point of this parabola. We can show that Newton's method converges according to

$$\|\mathbf{x}_{t+1} - \mathbf{x}^\star\| \leq c\|\mathbf{x}_t - \mathbf{x}^\star\|^2$$

when $\mathbf{x}_t - \mathbf{x}^\star$ is small enough. We can see here that the squared term provides very fast convergence once we are in the neighbourhood of the optimum. Newton's method can also be used to find a root of a function. Suppose $f : \mathbb{R} \to \mathbb{R}$, and define $f' = g$.

$$x_{t+1} = x_t - \frac{f'(x_t)}{f''(x_t)} = x_t - \frac{g(x_t)}{g'(x_t)}$$

So we can find the root of $g$ by computing the stationary point of $f$. We are essentially taking a linear approximation at a point, and setting this linear approximation to zero.

## 2.8 Barrier methods

Suppose we impose a constraint on an optimisation problem, for instance minimising $f(\mathbf{x})$ such that $f_i(\mathbf{x}) \leq 0$ for $1 \leq i \leq m$. We can transform such a constrained problem into an unconstrained problem. Let us minimise

$$f(\mathbf{x}) + \sum_{i=1}^{m} \phi(f_i(\mathbf{x}))$$

where $\phi(y_i) = +\infty$ outside the feasible set, and $\phi(y_i) = 0$ inside the feasible set. However, this $\phi$ function is not differentiable, so this introduces even more problems. We instead consider a *logarithmic barrier function*. Let us minimise the unconstrained problem

$$tf(\mathbf{x}) - \sum_{i=1}^{m} \log(-f_i(\mathbf{x})) \implies \phi(x) = -\log(-x)$$

This barrier function is infinite for negative $x$, and gradually rises as $x \to 0$. When $t$ is chosen to be very large, the optimum of this problem is very close to the optimum of the original problem.

**Algorithm 2:** Barrier Method

**Result:** Global minimum of $f(\mathbf{x})$
start at a point $\mathbf{x}$ inside the feasible set;
set $t$ to be a positive real number;
**repeat**
  solve the minimiser of $tf(\mathbf{x}) - \sum_{i=1}^{m} \log(-f_i(\mathbf{x}))$ with $\mathbf{x}$ as the initial point using Newton's method giving $\mathbf{x}^\star$;
  $\mathbf{x} \leftarrow \mathbf{x}^\star$;
  $t \leftarrow \alpha t$ for some fixed $\alpha > 1$;
**until** *$t$ is large enough*;

# 3 Lagrange multipliers

## 3.1 Introduction and Lagrange sufficiency

Consider minimising $f(\mathbf{x})$ subject to $\mathbf{x} \in \mathcal{X}$, $h(\mathbf{x}) = \mathbf{b}$ where $h : \mathbb{R}^n \to \mathbb{R}^m$. The *Lagrangian* associated with this problem is

$$L(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) - \boldsymbol{\lambda}^\intercal (h(\mathbf{x}) - \mathbf{b})$$

where $\boldsymbol{\lambda} \in \mathbb{R}^m$ is the vector of Lagrange multipliers. We want to instead minimise $L(\mathbf{x}, \boldsymbol{\lambda}), x \in \mathcal{X}$.

**Theorem** (Lagrange Sufficiency)**.** Suppose we can find a $\boldsymbol{\lambda}^\star$ such that
(i) $\min_{\mathbf{x}\in\mathcal{X}} L(\mathbf{x}, \boldsymbol{\lambda}^\star) = L(\mathbf{x}^\star, \boldsymbol{\lambda}^\star)$
(ii) $\mathbf{x}^\star \in \mathcal{X}(\mathbf{b}) = \{\mathbf{x} : \mathbf{x} \in \mathcal{X}, h(\mathbf{x}) = \mathbf{b}\}$
Then $\mathbf{x}^\star$ is optimal for the original constrained problem, i.e.

$$\min_{\mathbf{x}\in\mathcal{X}(\mathbf{b})} f(\mathbf{x}) = f(\mathbf{x}^\star)$$

*Proof.* First, note that condition (ii) states that $f(\mathbf{x}^\star) \geq \min_{\mathbf{x}\in\mathcal{X}(\mathbf{b})} f(\mathbf{x})$, because $\mathbf{x}^\star$ is feasible. Then,

$$\begin{aligned}
\min_{\mathbf{x}\in\mathcal{X}(\mathbf{b})} f(\mathbf{x}) &= \min_{\mathbf{x}\in\mathcal{X}(\mathbf{b})} f(\mathbf{x}) - \underbrace{(\boldsymbol{\lambda}^\star)^\intercal (h(\mathbf{x}) - \mathbf{b})}_{0 \text{ when } \mathbf{x}\in\mathcal{X}(\mathbf{b})} \\
&\geq \min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) - (\boldsymbol{\lambda}^\star)^\intercal (h(\mathbf{x}) - \mathbf{b}) \\
&= \min_{\mathbf{x}\in\mathcal{X}} L(\mathbf{x}, \boldsymbol{\lambda}^\star) \\
&= L(\mathbf{x}^\star, \boldsymbol{\lambda}^\star) \\
&= f(\mathbf{x}^\star) - (\boldsymbol{\lambda}^\star)^\intercal (h(\mathbf{x}^\star) - \mathbf{b}) \\
&= f(\mathbf{x}^\star)
\end{aligned}$$

□

**Example.**

$$\begin{aligned}
\underset{x\in\mathbb{R}^3}{\text{minimise}} \quad & -x_1 - x_2 + x_3 \\
\text{subject to} \quad & x_1^2 + x_2^2 = 4 \\
& x_1 + x_2 + x_3 = 1
\end{aligned}$$

In this problem, we have

$$h(\mathbf{x}) = \begin{pmatrix} x_1^2 + x_2^2 \\ x_1 + x_2 + x_3 \end{pmatrix}; \quad \mathbf{b} = \begin{pmatrix} 4 \\ 1 \end{pmatrix}$$

Taking Lagrange multipliers, we have

$$\begin{aligned}
L(\mathbf{x}, \boldsymbol{\lambda}) &= (-x_1 - x_2 + x_3) - \lambda_1(x_1^2 + x_2^2 - 4) - \lambda_2(x_1 + x_2 + x_3 - 1) \\
&= (-(1 + \lambda_2)x_1 - \lambda_1 x_1^2) + (-(1 + \lambda_2)x_2 - \lambda_1 x_2^2) + (1 - \lambda_2)x_3 + 4\lambda_1 + \lambda_2
\end{aligned}$$

We want to fix a value of $\boldsymbol{\lambda}$ and minimise $L$, only considering solutions such that $\mathbf{x}^\star$ is finite. Note that if $\lambda_1 > 0$, then the first bracket can be made as small as we like by picking very small values of $x_1$;

this bracket would diverge to negative infinity so we cannot choose such a $\lambda_1$. If $\lambda_2 \neq 1$, the infimum is also negative infinity by considering the $x_3$ term. So let us consider $\lambda_1 \leq 0, \lambda_2 = 1$. Setting the derivative of the first term to zero, we have

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}x_1}\left(-(1+\lambda_2)x_1 - \lambda_1 x_1^2\right) = -(1+\lambda_2) - 2\lambda_1 x_1 = 0 \\ \implies x_1 = \frac{-1-\lambda_2}{2\lambda_1} \\ = \frac{-2}{2\lambda_1} \\ = \frac{-1}{\lambda_1}\end{aligned}$$

Setting the derivative of the second term to zero,

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}x_1}\left(-(1+\lambda_2)x_2 - \lambda_1 x_2^2\right) = -(1+\lambda_2) - 2\lambda_1 x_2 = 0 \\ \implies x_2 = \frac{-1}{\lambda_1}\end{aligned}$$

We now want to choose $\lambda_1$ such that $x_1, x_2, x_3$ satisfy the constraints.

$$x_1^2 + x_2^2 = 4 \implies x_1^2 = x_2^2 = 2 \implies x_1 = x_2 = \sqrt{2}$$

Note that $x_1, x_2 > 0$ since $\lambda_1 \leq 0$, and correspondingly $\lambda_1 = \frac{-1}{\sqrt{2}}$. Further, we can now find $x_3 = 1 - 2\sqrt{2}$. This solution optimises the original problem.

## 3.2 Using Lagrange multipliers in general

Consider the problem

$$\begin{aligned}&\underset{\mathbf{x}\in\mathcal{X}}{\text{minimise}} && f(\mathbf{x}) \\ &\text{subject to} && h(\mathbf{x}) \leq \mathbf{b}\end{aligned}$$

We can solve this problem using the following steps.

(1) Add a slack variable $\mathbf{s}$ to transform the problem to

$$\begin{aligned}&\underset{\mathbf{x}\in\mathcal{X}}{\text{minimise}} && f(\mathbf{x}) \\ &\text{subject to} && h(\mathbf{x}) + \mathbf{s} = \mathbf{b} \\ & && \mathbf{s} \geq 0\end{aligned}$$

(2) Calculate the Lagrangian,

$$L(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{s}) = f(\mathbf{x}) - \boldsymbol{\lambda}^\intercal(h(\mathbf{x}) + \mathbf{s} - \mathbf{b})$$

(3) Let

$$\boldsymbol{\Lambda} = \left\{\boldsymbol{\lambda} : \inf_{\mathbf{x}\in\mathcal{X};\, \mathbf{s}\geq 0} L(\mathbf{x}, \mathbf{s}, \boldsymbol{\lambda}) > -\infty\right\}$$

(4) For each $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$, find $\mathbf{x}^\star(\boldsymbol{\lambda}), \mathbf{s}^\star(\boldsymbol{\lambda})$ such that

$$\min_{\mathbf{x}\in\mathcal{X};\, \mathbf{s}\geq 0} L(\mathbf{x}, \mathbf{s}, \boldsymbol{\lambda}) = L(\mathbf{x}^\star(\boldsymbol{\lambda}), \mathbf{s}^\star(\boldsymbol{\lambda}), \boldsymbol{\lambda})$$

(5) Find $\boldsymbol{\lambda}^\star \in \boldsymbol{\Lambda}$ such that $(\mathbf{x}^\star(\boldsymbol{\lambda}), \mathbf{s}^\star(\boldsymbol{\lambda}))$ is feasible, i.e.

$$h(\mathbf{x}^\star(\boldsymbol{\lambda}^\star)) = \mathbf{b}; \quad \mathbf{s}^\star(\boldsymbol{\lambda}^\star) \geq 0$$

## 3.3 Complementary slackness

In step (4) above, we want to minimise the Lagrangian, i.e.

$$\begin{aligned} \underset{\mathbf{x}\in\mathcal{X}}{\text{minimise}} \quad & f(\mathbf{x}) - \boldsymbol{\lambda}^\intercal(h(\mathbf{x}) - \mathbf{b}) - \boldsymbol{\lambda}^\intercal \mathbf{s} \\ \text{subject to} \quad & \mathbf{s} \geq 0 \end{aligned}$$

Suppose, for a particular value of $\boldsymbol{\lambda}$, that we solve this problem and arrive at $\mathbf{x}^\star(\boldsymbol{\lambda}), \mathbf{s}^\star(\boldsymbol{\lambda})$. Let

$$\boldsymbol{\lambda} = \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_m \end{pmatrix}$$

If $\lambda_i > 0$, then for some large $\mathbf{s}$ we can make $f \to -\infty$, hence $\boldsymbol{\lambda} \notin \boldsymbol{\Lambda}$. Hence, given $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$, we must have $\lambda_i \leq 0$. Now, if $\lambda_i < 0$ for some $i$, we would want to choose $s_i = 0$ to minimise the increase to the function caused by the slack variable. If $\lambda_i = 0$, then $s_i$ can be chosen arbitrarily since it will have no increase on the value of $f$. With these choices of $s_i$, we can make $\boldsymbol{\lambda}^\intercal \mathbf{s} = 0$, thus making the slack variable not impact the value of $f$. So either

- $h(\mathbf{x})_i = b_i$ and $\lambda_i \leq 0$, or
- $h(\mathbf{x})_i \geq b_i$ and $\lambda_i = 0$.

Alternatively (less precisely),

$$\lambda_i s_i = 0$$

In other words, either the constraint inequality is tight (defined by an equality) and the Lagrange multipliers are slack (defined by an inequality), or the constraint inequality is slack and the Lagrange multipliers are tight.

**Example.**

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^2}{\text{minimise}} \quad & x_1 - 3x_2 \\ \text{subject to} \quad & x_1^2 + x_2^2 \leq 4 \\ & x_1 + x_2 \leq 2 \end{aligned}$$

Adding slack variables, we have

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^2}{\text{minimise}} \quad & x_1 - 3x_2 \\ \text{subject to} \quad & x_1^2 + x_2^2 + s_1 = 4 \\ & x_1 + x_2 + s_2 = 2 \\ & s_1 \geq 0 \\ & s_2 \geq 0 \end{aligned}$$

Taking the Lagrangian,

$$\begin{aligned} L(\mathbf{x}, \mathbf{s}, \boldsymbol{\lambda}) &= (x_1 - 3x_2) - x_1(x_1^2 + x_2^2 + s_1 - 4) - \lambda_2(x_1 + x_2 + s_2 - 2) \\ &= \big((1 - \lambda_2)x_1 - \lambda_1 x_1^2\big) + \big((-3 - \lambda_2)x_2 - \lambda_1 x_2^2\big) - \lambda_1 s_1 - \lambda_2 s_2 + (4\lambda_1 + 2\lambda_2) \end{aligned}$$

We must have $\lambda_1, \lambda_2 \leq 0$ by considering the slack variable. By complementary slackness,

$$\lambda_1 s_1 = \lambda_2 s_2 = 0 \text{ at the optimum}$$

Minimising each term independently, we have

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

If $\lambda_1 = 0$, the above two equations are contradictory. Hence $\lambda_1 < 0$, giving $s_1 = 0$. If $\lambda_2 < 0$, then $s_2 = 0$ by complementary slackness, so

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

Solving the lower two equations give

$$(x_1, x_2) = (0, 2), (2, 0)$$

If $(x_1, x_2) = (0, 2)$, solving the first two equations gives $(\lambda_1, \lambda_2) = (1, -3)$ which is impossible since $\lambda_1$ must be negative. Similarly, if $(x_1, x_2) = (2, 0)$, solving the first two equations gives $(\lambda_1, \lambda_2) = (-1, 1)$ which is impossible again. We have ruled out every case apart from $\lambda_1 < 0, \lambda_2 = 0$. In this case,

$$\begin{aligned} 1 - 2\lambda_1 x_1 &= 0 \\ -3 - 2\lambda_1 x_2 &= 0 \\ x_1^2 + x_2^2 &= 4 \\ x_1 + x_2 + s_2 &= 2 \end{aligned}$$

The first two equations give

$$x_1 = \frac{1}{2\lambda_1}; \quad x_2 = \frac{-3}{2\lambda_1}$$

Substituting into the third equation,

$$\lambda_1^2 = \frac{5}{8} \implies \lambda_1 = -\sqrt{\frac{5}{8}}$$

Hence,

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

which is feasible using the fourth equation. By Lagrange sufficiency, this is the optimum for the original problem.

## 3.4 Weak duality

We would like to solve a problem

$$\begin{aligned} \underset{\mathbf{x}\in\mathcal{X}}{\text{minimise}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & h(\mathbf{x}) = \mathbf{b} \end{aligned}$$

by constructing the Lagrangian

$$L(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) - \boldsymbol{\lambda}^\intercal (h(\mathbf{x}) - \mathbf{b})$$

We now define the quantity

$$g(\boldsymbol{\lambda}) = \inf_{\mathbf{x}\in\mathcal{X}} L(\mathbf{x}, \boldsymbol{\lambda})$$

**Theorem** (Weak duality theorem). If $\mathbf{x} \in \mathcal{X}(\mathbf{b})$ and $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$, then $f(\mathbf{x}) \geq g(\boldsymbol{\lambda})$. In particular,

$$\inf_{\mathbf{x}\in\mathcal{X}(\mathbf{b})} f(\mathbf{x}) \geq \sup_{\boldsymbol{\lambda}\in\boldsymbol{\Lambda}} g(\boldsymbol{\lambda})$$

*Proof.*

$$\begin{aligned} \inf_{\mathbf{x}\in\mathcal{X}(\mathbf{b})} f(\mathbf{x}) &= \inf_{\mathbf{x}\in\mathcal{X}(\mathbf{b})} f(\mathbf{x}) - \boldsymbol{\lambda}^\intercal (h(\mathbf{x}) - \mathbf{b}) \\ &\geq \inf_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) - \boldsymbol{\lambda}^\intercal (h(\mathbf{x}) - \mathbf{b}) \\ &= \inf_{\mathbf{x}\in\mathcal{X}} L(\mathbf{x}, \boldsymbol{\lambda}) \\ &= g(\boldsymbol{\lambda}) \end{aligned}$$

□

Using this weak duality property, if $\inf_{\mathbf{x}\in\mathcal{X}(\mathbf{b})} f(\mathbf{x})$ is difficult to solve, we can first attempt $\sup_{\boldsymbol{\lambda}\in\boldsymbol{\Lambda}} g(\boldsymbol{\lambda})$. The problem

$$\begin{aligned} \text{maximise} \quad & g(\boldsymbol{\lambda}) \\ \text{subject to} \quad & \boldsymbol{\lambda} \in \boldsymbol{\Lambda} \end{aligned}$$

is called the *dual problem*. The original is called the *primal problem*. The optimal cost of the primal problem is always greater than or equal to the optimal cost of the dual problem. The *duality gap* is the difference:

$$\inf_{\mathbf{x}\in\mathcal{X}(\mathbf{b})} f(\mathbf{x}) - \sup_{\boldsymbol{\lambda}\in\boldsymbol{\Lambda}} g(\boldsymbol{\lambda})$$

If the duality gap is zero, then we say that *strong duality* holds. This strengthens the inequality into an equality.

## 3.5 Strong duality and the Lagrange method

If the Lagrange method works, then we know that

$$\inf_{\mathbf{x}\in\mathcal{X}} L(\mathbf{x}, \boldsymbol{\lambda}) = \inf_{\mathbf{x}\in\mathcal{X}(\mathbf{b})} L(\mathbf{x}, \boldsymbol{\lambda})$$

So, taking such a $\boldsymbol{\lambda}$ in the proof above, we have equality instead of inequality. Hence the problem has strong duality. Conversely, if the duality gap is zero, then there exists a $\boldsymbol{\lambda}$ such that the inequality above is an equality. Hence, for this $\boldsymbol{\lambda}$,

$$\inf_{\mathbf{x}\in\mathcal{X}} L(\mathbf{x},\boldsymbol{\lambda}) = \inf_{\mathbf{x}\in\mathcal{X}(\mathbf{b})} L(\mathbf{x},\boldsymbol{\lambda})$$

Hence this is the $\boldsymbol{\lambda}$ which will solve the Lagrange method. In summary, strong duality holds exactly when the Lagrange method works.

## 3.6 Hyperplane condition for strong duality

**Definition.** A function $\phi : \mathbb{R}^m \to \mathbb{R}$ is said to have a *supporting hyperplane* at a point $\mathbf{b}$ if there exists $\boldsymbol{\lambda} \in \mathbb{R}^m$ such that for all $\mathbf{c} \in \mathbb{R}^m$,

$$\phi(\mathbf{c}) \geq \phi(\mathbf{b}) + \boldsymbol{\lambda}^\intercal(\mathbf{c}-\mathbf{b})$$

Pictorially, $\phi$ has a supporting hyperplane if there is a plane passing through $(\mathbf{b}, \phi(\mathbf{b}))$, where $\phi$ is always above the plane. This could be, for example, a tangent plane at $\mathbf{b}$.

**Definition.** We define a function $\phi : \mathbb{R}^m \to \mathbb{R}$ associated with the primal problem by

$$\phi(\mathbf{c}) = \inf_{\mathbf{x}\in\mathcal{X}(\mathbf{c})} f(\mathbf{x})$$

This $\phi$ can be thought of as the optimal cost of a family of optimisation problems with different functional constraint values $\mathbf{c}$. This is called the *value function*.

**Theorem** (Strong duality theorem)**.** Strong duality holds if and only if the value function $\phi$ has a supporting hyperplane at $\mathbf{b}$.

*Proof.* First, we show that a supporting hyperplane implies strong duality. We have $\boldsymbol{\lambda}$ such that

$$\phi(\mathbf{c}) \geq \phi(\mathbf{b}) + \boldsymbol{\lambda}^\intercal(\mathbf{c}-\mathbf{b})$$

Then, we have

$$\begin{aligned} g(\boldsymbol{\lambda}) &= \inf_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) - \boldsymbol{\lambda}^\intercal(h(\mathbf{x})-\mathbf{b}) \\ &= \inf_{\mathbf{c}} \inf_{\mathbf{x}\in\mathcal{X}(\mathbf{c})} f(\mathbf{x}) - \underbrace{\boldsymbol{\lambda}^\intercal(h(\mathbf{x})-\mathbf{c})}_{\text{zero since we are extremising}} - \boldsymbol{\lambda}^\intercal(\mathbf{c}-\mathbf{b}) \\ &= \inf_{\mathbf{c}} \phi(\mathbf{c}) - \boldsymbol{\lambda}^\intercal(\mathbf{c}-\mathbf{b}) \\ &\geq \phi(\mathbf{b}) \end{aligned}$$

By weak duality, we also have the reverse direction: $g(\boldsymbol{\lambda}) \leq \phi(\mathbf{b})$ Hence, $g(\boldsymbol{\lambda}) = \phi(\mathbf{b})$ and strong duality holds. Conversely, if strong duality holds, we want to show the existence of such a hyperplane.

We have We have $\boldsymbol{\lambda}$ such that $g(\boldsymbol{\lambda}) = \phi(\mathbf{b})$. For such a $\boldsymbol{\lambda}$, we have

$$\begin{aligned}\phi(\mathbf{b}) = g(\boldsymbol{\lambda}) &= \inf_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) - \boldsymbol{\lambda}^\intercal(h(\mathbf{x}) - \mathbf{b}) \\ &= \inf_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) - \boldsymbol{\lambda}^\intercal(h(\mathbf{x}) - \mathbf{c}) - \boldsymbol{\lambda}^\intercal(\mathbf{c} - \mathbf{b}) \\ &\leq \phi(\mathbf{c}) - \boldsymbol{\lambda}^\intercal(\mathbf{c} - \mathbf{b})\end{aligned}$$

The last inequality holds due to weak duality. So $\boldsymbol{\lambda}$ gives a supporting hyperplane. □

## 3.7 Strong duality and convex functions

We would now like to consider for which problems $\phi(\mathbf{b})$ has a supporting hyperplane. The following theorem is stated without proof.

**Theorem.** A function $\phi : \mathbb{R}^m \to \mathbb{R}$ is convex if and only if every point $\mathbf{b} \in \mathbb{R}^m$ has a supporting hyperplane.

Now, for which problems do we have a convex value function?

**Theorem.** Consider a minimisation problem

$$\begin{aligned}\underset{\mathbf{x}\in\mathcal{X}}{\text{minimise}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & h(\mathbf{x}) \leq \mathbf{b}\end{aligned}$$

with value function $\phi$. Then $\phi$ is convex if:
(i) $\mathcal{X}$ is convex;
(ii) $f$ is convex;
(iii) $h$ is convex.

This is proven in the example sheets.

## 3.8 Shadow prices interpretation of Lagrange multipliers

Suppose a factory owner produces $n$ types of products from $m$ types of raw materials. Suppose the owner produces $\mathbf{x} = (x_1, x_2, \dots, x_n)$ products, then the profit is some function $f(\mathbf{x})$. We then create $h_j(\mathbf{x})$ to be the amount of raw material $j$ consumed when making products $\mathbf{x}$. The owner wants to maximise $f(\mathbf{x})$ subject to $h_i(\mathbf{x}) \leq b_i$ where $b_i$ is the maximum amount of raw material $i$ that is available.

Now, suppose a supplier offers some $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \dots, \varepsilon_m)$ extra raw materials to the factory owner. We would like to calculate how much this $\boldsymbol{\varepsilon}$ is worth. The factory owner will try to maximise this new problem, replacing $\mathbf{b} \mapsto \mathbf{b} + \boldsymbol{\varepsilon}$. For a small enough $\boldsymbol{\varepsilon}$, this can be expressed easily using the value function.

$$\phi(\mathbf{b} + \boldsymbol{\varepsilon}) - \phi(\mathbf{b}) \approx \sum_{j=1}^{m} \frac{\partial \phi}{\partial b_j} \varepsilon_j$$

The quantity $\frac{\partial \phi}{\partial b_j}$ is the price of material $j$, and $\nabla\phi(\mathbf{b})$ is the vector of prices. These are called the 'shadow prices'; they are hidden to the outside world but depend on the internal state of the factory.

**Theorem.** If $\phi$ is differentiable at $\mathbf{b}$ and has a supporting hyperplane given by $\boldsymbol{\lambda}$, then

$$\boldsymbol{\lambda} = \nabla\phi(\mathbf{b})$$

*Proof.* Let $\mathbf{a} = (a_1, a_2, \dots, a_m)$ be an arbitrary vector. Then from the supporting hyperplane condition, for some small $\delta > 0$ we have

$$\frac{\phi(\mathbf{b} + \delta\mathbf{a})}{\delta} \geq \boldsymbol{\lambda}^\intercal \mathbf{a}$$

Since $\phi$ is differentiable, the limit can be taken to give

$$\nabla\phi(\mathbf{b}) \cdot \mathbf{a} \geq \boldsymbol{\lambda}^\intercal \mathbf{a}$$

But $\mathbf{a}$ was arbitrary. This can only hold if $\boldsymbol{\lambda} = \nabla\phi(\mathbf{b})$ as required. So the Lagrange multiplier $\boldsymbol{\lambda}$ at $\mathbf{b}$ is equal to the gradient vector of $\phi$ which is the gradient of partial derivatives and also the vector of shadow prices. □

Suppose that a particular raw material was not used up. Then there is a slack value in the inequality. The shadow price is zero in this instance, since we do not need more of this material. So the corresponding Lagrange multiplier is equal to zero. Conversely, if we are paying something for this material, then we must have used up all of that material. This is exactly the complementary slackness property seen earlier.

There is also an economics interpretation of the dual problem. Such a problem can be seen from the perspective of the raw material seller. This seller charges a certain price $\boldsymbol{\lambda}$ for their raw materials, and then buys the finished product from the factory. The profit of the raw material seller is

$$\underbrace{\boldsymbol{\lambda}^\intercal(h(\mathbf{x}) - \mathbf{b})}_{\text{cost of materials}} - \underbrace{f(\mathbf{x})}_{\text{buying products}}$$

For every choice of $\boldsymbol{\lambda}$, the factory owner will try to maximise their profit, that is, find an $\mathbf{x}^\star$ such that we maximise

$$\underbrace{f(\mathbf{x})}_{\text{selling products}} - \underbrace{\boldsymbol{\lambda}^\intercal(h(\mathbf{x}) - \mathbf{b})}_{\text{cost of materials}}$$

# 4 Linear programming

## 4.1 Linear programs

A linear program is a specific case of a constrained optimisation problem in which the objective function and all constraints are linear functions. For instance, consider the problem

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^4}{\text{minimise}} \quad & 2x_1 - x_2 + 4x_3 \\ \text{subject to} \quad & x_1 + x_2 + x_4 \leq 2 \\ & 3x_2 - x_3 = 5 \\ & x_3 + x_4 \geq 3 \\ & x_1 \geq 0 \\ & x_3 \leq 0 \end{aligned}$$

A general linear program is of the form

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimise}} \quad & \mathbf{c}^\intercal \mathbf{x} \\ \text{subject to} \quad & \mathbf{a}_i^\intercal \mathbf{x} \geq b_i, i \in M_1 \\ & \mathbf{a}_i^\intercal \mathbf{x} \leq b_i, i \in M_2 \\ & \mathbf{a}_i^\intercal \mathbf{x} = b_i, i \in M_3 \\ & x_j \geq 0, j \in N_1 \\ & x_j \leq 0, j \in N_2 \end{aligned}$$

Note that we can convert the first inequalities to the other direction by inverting the sign of $\mathbf{a}$. We can convert the 'sign' constraints (the last two constraints) by letting $\mathbf{a}$ be a one-hot vector, thus writing them in terms of the first two inequality types. We call this process *reduction* to an equivalent form. Two linear programs are *equivalent* if any feasible solution for one problem can be converted into a feasible solution for the other, with the same cost. We can reduce any linear problem into the form

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimise}} \quad & \mathbf{c}^\intercal \mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} \geq \mathbf{b} \end{aligned}$$

where

$$A = \begin{pmatrix} \cdots & \mathbf{a}_1^\intercal & \cdots \\ \cdots & \mathbf{a}_2^\intercal & \cdots \\ & \vdots & \\ \cdots & \mathbf{a}_m^\intercal & \cdots \end{pmatrix}; \quad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}$$

This is known as the *general form* of a linear programming problem. We could alternatively use a 'less-than' inequality, or simply an equality using a slack variable vector. A linear problem is said to

be in *standard form* if it is written as

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimise}} \quad & \mathbf{c}^\intercal \mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

This is a special case of the general form. However, we can always reduce any general-form problem into a standard-form problem. First, we add slack variables to convert the inequality into an equality. Then we can convert each variable $x_i$ into the sum of $x_j^+ - x_j^-$, where $x_j^+, x_j^- \geq 0$. Then, we have the problem

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimise}} \quad & \mathbf{c}^\intercal(\mathbf{x}^+ - \mathbf{x}^-) \\ \text{subject to} \quad & A(\mathbf{x}^+ - \mathbf{x}^-) = \mathbf{b} \\ & \mathbf{x}^+, \mathbf{x}^- \geq 0 \end{aligned}$$

Then by concatenating the vectors $\mathbf{x}^+, \mathbf{x}^-$ into a larger vector $\mathbf{z} \in [0,\infty)^{2n}$, we have the standard form as required.

## 4.2 Maximising convex functions

Solving linear programs can be seen as a special case of maximising a convex function, since we can maximise $\mathbf{c}^\intercal \mathbf{x}$. Consider the problem

$$\begin{aligned} \text{minimise} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & x \in C, C \text{ convex} \\ & \mathbf{x} \geq 0 \end{aligned}$$

where $f$ is a convex function. Since $C$ is convex, if $\mathbf{z} = (1-\lambda)\mathbf{x} + \lambda\mathbf{y}$ we have

$$f(\mathbf{z}) \leq (1-\lambda)f(\mathbf{x}) + \lambda f(\mathbf{y}) \leq \max\{f(\mathbf{x}), f(\mathbf{y})\}$$

If we wish to maximise $f$ over $C$, we might guess that we only need to consider points on the boundary. After all, any point not on the boundary can be written as the weighted average of two points on the boundary. Considering those points will give a greater (or equal) value for $f$.

**Definition.** A point $\mathbf{x}$ in a convex set $C$ is an *extreme point* if it cannot be written as a convex combination of two distinct points in $C$; that is,

$$(1-\delta)\mathbf{y} + \delta\mathbf{z}$$

for $\delta \in (0,1)$ and $\mathbf{y} \neq \mathbf{z}$.

So, more precisely, convex functions on convex sets are maximised at extreme points.

## 4.3 Basic solutions and basic feasible solutions

Consider a linear problem in standard form.

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimise}} \quad & \mathbf{c}^\intercal\mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

where $A \in \mathbb{R}^{m\times n}, \mathbf{x} \in \mathbb{R}^n, \mathbf{b} \in \mathbb{R}^m$.

**Definition.** A vector $\mathbf{x}$ is said to be a *basic solution* if it satisfies $A\mathbf{x} = \mathbf{b}$ (that is, it is a solution) and $\mathbf{x}$ has at most $m$ nonzero entries. If also the nonzero entries are positive, then this is called a *basic feasible solution*, since it lies in the feasible set $\mathbf{x} \geq 0$.

We will start the analysis of basic solutions by making three assumptions (one is defined later).

A: All $m$ rows of $A$ are linearly independent. That is, $\{\mathbf{a}_1^\intercal, \dots, \mathbf{a}_m^\intercal\}$ is a linearly independent set. This assumption can be made without loss of generality since we can simply remove linearly dependent constraints.

B: Every set of $m$ columns of $A$ is linearly independent. That is, any $m$-subset of the set of columns $\{A_1, \dots, A_n\}$ is a linearly independent set. This can also be made without loss of generality by removing the linearly dependent variables.

To find a basic solution, we will start by choosing the coordinates $B(1), B(2), \dots, B(m)$ to be the indices of $\mathbf{x}$ that are allowed to be nonzero. Now, $A\mathbf{x}$ is

$$\begin{pmatrix} \vdots & & \vdots \\ A_1 & \cdots & A_n \\ \vdots & & \vdots \end{pmatrix} \begin{pmatrix} 0 \\ \vdots \\ x_{B(1)} \\ \vdots \\ x_{B(2)} \\ \vdots \\ x_{B(m)} \\ \vdots \\ 0 \end{pmatrix} = \underbrace{\begin{pmatrix} \vdots & & \vdots \\ A_{B(1)} & \cdots & A_{B(m)} \\ \vdots & & \vdots \end{pmatrix}}_{B} \begin{pmatrix} x_{B(1)} \\ x_{B(2)} \\ \vdots \\ x_{B(m)} \end{pmatrix}$$

By setting $A\mathbf{x} = \mathbf{b}$, using the above assumptions, we can invert the matrix on the left-hand side to get

$$\begin{pmatrix} x_{B(1)} \\ x_{B(2)} \\ \vdots \\ x_{B(m)} \end{pmatrix} = \begin{pmatrix} \vdots & & \vdots \\ A_{B(1)} & \cdots & A_{B(m)} \\ \vdots & & \vdots \end{pmatrix}^{-1} \mathbf{b} = B^{-1}\mathbf{b}$$

We call $B$ the basis matrix. The indices $x_{B(1)}, \dots, x_{B(m)}$ are called the basic variables. The indices $B(i)$ are called the basic indices. The columns $A_{B(i)}$ are called the basic columns. If $B^{-1}\mathbf{b} \geq 0$, we have found a basic feasible solution. We now need to specify one further assumption in order to continue to analyse basic solutions.

C: Every basic solution has *exactly* $m$ nonzero entries. This assumption is known as the non-degeneracy assumption. This assumption cannot be created without loss of generality, but it is far simpler to discuss problems with this assumption met. Throughout this course, we will keep this assumption to be true.

## 4.4 Extreme points of the feasible set in standard form

Consider a linear program in standard form.

**Theorem.** $\mathbf{x}$ is an extreme point (of the set $\{\mathbf{x} : A\mathbf{x} = \mathbf{b}, \mathbf{x} \geq 0\}$) if and only if $\mathbf{x}$ is a basic feasible solution.

*Remark.* Since a linear program is optimised at extreme points, we only need to consider the basic feasible solutions in order to solve the original problem. We will pick all possible $m$ columns of $A$ (there are $\binom{n}{m}$ such choices) to find all basic solutions. Filter to consider only basic feasible solutions, then evaluate $\mathbf{c}^\intercal\mathbf{x}$ to find the $\mathbf{x}$ which has the least cost. This algorithm will always work, but the amount of choices to evaluate in higher dimensions becomes too inefficient for real-world use.

*Proof.* First, suppose we know $\mathbf{x}$ is a basic feasible solution, and there exist feasible $\mathbf{y}, \mathbf{z}$ such that $\mathbf{x} = (1-\delta)\mathbf{y} + \delta\mathbf{z}$ and $\delta \in (0,1)$. We know $\mathbf{x} \geq 0$ and $\mathbf{x}$ has at most $m$ nonzero entries. Since $\mathbf{y}, \mathbf{z}$ are positive, then $\mathbf{y}, \mathbf{z}$ must be zero in every index that $\mathbf{x}$ must be zero. Specifically, $y_j = z_j = 0$ for $j \notin \{B(1), \dots, B(m)\}$. Now, we define

$$\mathbf{y}_B = \begin{pmatrix} y_{B(1)} \\ \vdots \\ y_{B(m)} \end{pmatrix}; \quad \mathbf{z}_B = \begin{pmatrix} z_{B(1)} \\ \vdots \\ z_{B(m)} \end{pmatrix}$$

We then have $B\mathbf{y}_B = \mathbf{b}; B\mathbf{z}_B = \mathbf{b}$ because $A\mathbf{y} = A\mathbf{z} = \mathbf{b}$. Hence, $\mathbf{y}_B = \mathbf{z}_B = B^{-1}\mathbf{b}$ and so $\mathbf{x} = \mathbf{y} = \mathbf{z}$.

Conversely, suppose $\mathbf{x}$ is not a basic feasible solution. We wish to show it is not an extreme point. Then $\mathbf{x}$ has an amount of nonzero indices greater than $m$. Let such indices be $i_1, \dots, i_r$ where $r > m$. Consider the columns $A_{i_1}, \dots, A_{i_r}$. Since the rank of $A$ is only $m$, these columns form a linearly dependent set. Hence, we can find some weights, not all of which are zero, which give zero when multiplied by the columns.

$$w_{i_1}A_{i_1} + w_{i_2}A_{i_2} + \cdots + w_{i_r}A_{i_r} = 0$$

We now define the vector $\mathbf{w}$ by

$$w_i = \begin{cases} 0 & i \notin \{i_1, \dots, i_r\} \\ w_{i_j} & i = i_j \end{cases}$$

So we have a nonzero vector $\mathbf{w}$ with $A\mathbf{w} = 0$. We can consider the two points $\mathbf{x} \pm \varepsilon\mathbf{w}$, which satisfy $A(\mathbf{x} \pm \varepsilon\mathbf{w}) = 0$. Such perturbed points only change the nonzero indices of $\mathbf{x}$. So we can find an $\varepsilon$ small enough such that both of $\mathbf{x} \pm \varepsilon\mathbf{w}$ are in the feasible set, that is, $\mathbf{x} \pm \varepsilon\mathbf{w} \geq 0$. We therefore can express $\mathbf{x}$ as the midpoint of these two points, hence $\mathbf{x}$ is not an extreme point. □

# 5 Duality in linear programming

## 5.1 Strong duality of linear programs

**Theorem.** If a linear program is bounded and feasible, then strong duality holds.

*Proof.* This is true since the value function is convex. □

## 5.2 Duals of linear programs in standard form

Consider a linear program in standard form:

$$\begin{aligned} &\text{minimise} && \mathbf{c}^\intercal \mathbf{x} \\ &\text{subject to} && A\mathbf{x} = \mathbf{b} \\ & && \mathbf{x} \geq 0 \end{aligned}$$

The dual problem is therefore

$$\begin{aligned} &\text{maximise} && g(\boldsymbol{\lambda}) = \inf_{\mathbf{x} \in \mathcal{X}} L(\mathbf{x}, \boldsymbol{\lambda}) \\ &\text{subject to} && \boldsymbol{\lambda} \in \boldsymbol{\Lambda} \end{aligned}$$

The function $g$ is given by

$$\begin{aligned} g(\boldsymbol{\lambda}) &= \inf_{\mathbf{x} \geq 0} \mathbf{c}^\intercal \mathbf{x} - \boldsymbol{\lambda}^\intercal (A\mathbf{x} - \mathbf{b}) \\ &= \inf_{\mathbf{x} \geq 0} (\mathbf{c}^\intercal - \boldsymbol{\lambda}^\intercal A)\mathbf{x} + \boldsymbol{\lambda}^\intercal \mathbf{b} \end{aligned}$$

This is only bounded below where $\mathbf{c}^\intercal - \boldsymbol{\lambda}^\intercal A \geq 0$. Hence

$$\boldsymbol{\Lambda} = \{\boldsymbol{\lambda} : \ \boldsymbol{\lambda}^\intercal A \leq \mathbf{c}^\intercal\}$$

Further, the minimum value of $g$ for $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$ is $\boldsymbol{\lambda}^\intercal \mathbf{b}$. Therefore, the dual problem is

$$\begin{aligned} &\text{maximise} && \boldsymbol{\lambda}^\intercal \mathbf{b} \\ &\text{subject to} && \boldsymbol{\lambda}^\intercal A \leq \mathbf{c}^\intercal \end{aligned}$$

The dual of a linear program in standard form is a linear problem, but no longer in standard form.

## 5.3 Duals of linear programs in general form

Consider a linear program in general form:

$$\begin{aligned} &\text{minimise} && \mathbf{c}^\intercal \mathbf{x} \\ &\text{subject to} && A\mathbf{x} \geq \mathbf{b} \end{aligned}$$

We can introduce a slack variable $\mathbf{s}$ and write equivalently

$$\begin{aligned} &\text{minimise} && \mathbf{c}^\intercal \mathbf{x} \\ &\text{subject to} && A\mathbf{x} - \mathbf{s} = \mathbf{b} \\ & && \mathbf{s} \geq 0 \end{aligned}$$

To calculate the dual, we need to calculate $g(\boldsymbol{\lambda})$.

$$\begin{aligned} g(\boldsymbol{\lambda}) &= \inf_{\mathbf{x},\mathbf{s} \geq 0} \mathbf{c}^\intercal \mathbf{x} - \boldsymbol{\lambda}^\intercal (A\mathbf{x} - \mathbf{s} - \mathbf{b}) \\ &= \inf_{\mathbf{x},\mathbf{s} \geq 0} (\mathbf{c}^\intercal - \boldsymbol{\lambda}^\intercal A)\mathbf{x} + \boldsymbol{\lambda}^\intercal \mathbf{s} + \boldsymbol{\lambda}^\intercal \mathbf{b} \end{aligned}$$

In this case, since $\mathbf{x}$ may be any value, we must have $\mathbf{c}^\intercal - \boldsymbol{\lambda}^\intercal A = 0$. Further, since the slack variable can be any positive value, $\boldsymbol{\lambda}^\intercal \geq 0$. The infimum is $\boldsymbol{\lambda}^\intercal \mathbf{b}$ since $\mathbf{s}$ may be set to zero. Thus, the dual is

$$\begin{aligned} &\text{maximise} && \boldsymbol{\lambda}^\intercal \mathbf{b} \\ &\text{subject to} && \boldsymbol{\lambda}^\intercal A = \mathbf{c}^\intercal \\ & && \boldsymbol{\lambda} \geq 0 \end{aligned}$$

The dual of a general linear program is a linear program in standard form.

## 5.4 Dual of dual program

The dual of a dual problem is the primal problem. Suppose the primal problem is in standard form:

$$\begin{aligned} &\text{minimise} && \mathbf{c}^\intercal \mathbf{x} \\ &\text{subject to} && A\mathbf{x} = \mathbf{b} \\ & && \mathbf{x} \geq 0 \end{aligned}$$

We know the dual is

$$\begin{aligned} &\text{maximise} && \boldsymbol{\lambda}^\intercal \mathbf{b} \\ &\text{subject to} && \boldsymbol{\lambda}^\intercal A \leq \mathbf{c}^\intercal \end{aligned}$$

Equivalently,

$$\begin{aligned} -\ &\text{minimise} && -\boldsymbol{\lambda}^\intercal \mathbf{b} \\ &\text{subject to} && -\boldsymbol{\lambda}^\intercal A \geq -\mathbf{c}^\intercal \end{aligned}$$

Defining $\tilde{\boldsymbol{\lambda}} = -\boldsymbol{\lambda}^\intercal$, we have

$$\begin{aligned} -\ &\text{minimise} && \tilde{\boldsymbol{\lambda}} \mathbf{b} \\ &\text{subject to} && \tilde{\boldsymbol{\lambda}} A \geq -\mathbf{c}^\intercal \end{aligned}$$

We can find the dual of this problem using the solution above.

$$\begin{aligned} -\ &\text{maximise} && -\theta^\intercal \mathbf{c} \\ &\text{subject to} && \theta^\intercal A^\intercal = \mathbf{b}^\intercal \\ & && \theta \geq 0 \end{aligned}$$

This is equivalent to the primal problem.

## 5.5 Dual of arbitrary linear program

Consider the problem

$$\begin{array}{lll} \text{minimise} & \mathbf{c}^\intercal \mathbf{x} & \\ \text{subject to} & \mathbf{a}_i^\intercal \mathbf{x} \geq \mathbf{b}_i & i \in M_1 \\ & \mathbf{a}_i^\intercal \mathbf{x} \leq \mathbf{b}_i & i \in M_2 \\ & \mathbf{a}_i^\intercal \mathbf{x} = \mathbf{b}_i & i \in M_3 \\ & x_j \geq 0 & j \in N_1 \\ & x_j \leq 0 & j \in N_2 \\ & x_j \text{ free} & j \in N_3 \end{array}$$

The dual of this problem is

$$\begin{array}{lll} \text{maximise} & \mathbf{p}^\intercal \mathbf{b} & \\ \text{subject to} & p_i \geq 0 & i \in M_1 \\ & p_i \leq 0 & i \in M_2 \\ & p_i \text{ free} & i \in M_3 \\ & \mathbf{p}^\intercal \mathbf{A}_j \leq \mathbf{c}_j & j \in N_1 \\ & \mathbf{p}^\intercal \mathbf{A}_j \geq \mathbf{c}_j & j \in N_2 \\ & \mathbf{p}^\intercal \mathbf{A}_j = \mathbf{c}_j & j \in N_3 \end{array}$$

This will be shown in the example sheets.

## 5.6 Optimality conditions

If $\mathbf{x}$ is feasible for the primal, $\mathbf{p}$ is feasible for the dual, and complementary slackness holds, then $\mathbf{x}$ is optimal for the primal and $\mathbf{p}$ is optimal for the dual.

**Theorem** (Fundamental Theorem of Linear Programming)**.** Let $\mathbf{x}, \mathbf{p}$ be feasible solutions to the primal and dual problems respectively. Then $\mathbf{x}, \mathbf{p}$ are optimal for these problems if and only if

- $p_i(\mathbf{a}_i^\intercal \mathbf{x} - b_i) = 0$ for all $i$, and
- $(c_j - \mathbf{p}^\intercal \mathbf{A}_j)x_j = 0$ for all $j$.

*Proof.* First, let us define $u_i = p_i(\mathbf{a}_i^\intercal \mathbf{x} - b_i)$ and $v_j = (c_j - \mathbf{p}^\intercal \mathbf{A}_j)x_j$. Observe that if $\mathbf{x}, \mathbf{p}$ are feasible, then $u_i \geq 0$ for all $i$, and $v_j \geq 0$ for all $j$. This can be seen by the signs of the constraints on the primal and dual problems. Now,

$$\sum u_i = \sum p_i(\mathbf{a}_i^\intercal \mathbf{x} - b_i) = \mathbf{p}^\intercal A\mathbf{x} - \mathbf{p}^\intercal \mathbf{b}$$

Similarly,

$$\sum v_j = \sum (c_j - \mathbf{p}^\intercal \mathbf{A}_j)x_j = \mathbf{c}^\intercal \mathbf{x} - \mathbf{p}^\intercal A\mathbf{x}$$

Then,

$$\sum u_i + \sum v_j = \mathbf{c}^\intercal \mathbf{x} - \mathbf{p}^\intercal \mathbf{b}$$

which is the difference between the two objective functions in the primal and the dual. Hence,

$$0 \leq \sum u_i + \sum v_j = \mathbf{c}^\intercal \mathbf{x} - \mathbf{p}^\intercal \mathbf{b}$$

So if complementary slackness holds, then $u_i = 0$ and $v_j = 0$ for all $i, j$. This then implies that $\mathbf{c}^\intercal \mathbf{x} = \mathbf{p}^\intercal \mathbf{b}$. By weak duality, $\mathbf{x}$ and $\mathbf{p}$ must be optimal. Conversely, suppose $\mathbf{x}, \mathbf{p}$ are optimal. By strong duality, $\mathbf{c}^\intercal \mathbf{x} = \mathbf{p}^\intercal \mathbf{b}$.

$$0 \leq \sum u_i + \sum v_j = \mathbf{c}^\intercal \mathbf{x} - \mathbf{p}^\intercal \mathbf{b} = 0$$

Thus $\sum u_i + \sum v_j = 0$. Since all $u_i, v_j$ are non-negative, $u_i = 0$ and $v_j = 0$ for all $i, j$. Equivalently, complementary slackness holds. □

# 6 Simplex method

## 6.1 Introduction

Consider the problem

$$\begin{array}{ll} \text{minimise} & \mathbf{c}^\intercal \mathbf{x} \\ \text{subject to} & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{array}$$

The dual problem is

$$\begin{array}{ll} \text{maximise} & \boldsymbol{\lambda}^\intercal \mathbf{b} \\ \text{subject to} & \boldsymbol{\lambda}^\intercal A \leq \mathbf{c}^\intercal \end{array}$$

The optimality conditions are

- (primal feasibility) $A\mathbf{x} = \mathbf{b}; \mathbf{x} \geq 0$
- (dual feasibility) $A^\intercal \boldsymbol{\lambda} \leq \mathbf{c}$
- (complementary slackness) $\mathbf{x}^\intercal (\mathbf{c} - A^\intercal \boldsymbol{\lambda}) = 0$

Suppose $\mathbf{x}$ is a basic feasible solution given by

$$\mathbf{x}_B = (x_{B(1)}, \dots, x_{B(m)})$$

Substituting this $\mathbf{x}$ into the complementary slackness equation gives

$$\mathbf{x}_B^\intercal \mathbf{c}_B - \mathbf{x}_B^\intercal B^\intercal \boldsymbol{\lambda} = 0 \implies \mathbf{x}_B^\intercal (\mathbf{c}_B - B^\intercal \boldsymbol{\lambda}) = 0$$

For a basic feasible solution, $\mathbf{x}_B > 0$. Hence,

$$\mathbf{c}_B - B^\intercal \boldsymbol{\lambda} = 0$$

Hence

$$\boldsymbol{\lambda} = (B^\intercal)^{-1} \mathbf{c}_B$$

So for this $\mathbf{x}$ and this calculated $\boldsymbol{\lambda}$, primal feasibility and complementary slackness both hold. What remains now is to check if dual feasibility holds. Equivalently,

$$A^\intercal \boldsymbol{\lambda} \leq \mathbf{c} \implies A^\intercal (B^\intercal)^{-1} \mathbf{c}_B \leq \mathbf{c}$$

If this holds, then the optimality conditions are met. This means that we do not even need to explicitly find $\boldsymbol{\lambda}$ in order to check optimality; it suffices to check whether this single inequality holds. We define

$$\overline{\mathbf{c}} = \mathbf{c} - A^\intercal (B^\intercal)^{-1} \mathbf{c}_B$$

This is called the *vector of reduced costs*. Then the inequality $\overline{\mathbf{c}} \geq 0$ implies $\mathbf{x}$ is optimal.

## 6.2 Feasibility of basic directions

**Definition.** Let $P = \{\mathbf{x} : A\mathbf{x} = \mathbf{b}, \mathbf{x} \geq 0\}$ be the feasible set of a problem in standard form. Further, let $\mathbf{x} \in P$. A vector $\mathbf{d} \in \mathbb{R}^n$ is called a *feasible direction* if there exists $\theta > 0$ such that $\mathbf{x} + \theta\mathbf{d} \in P$.

Let $\mathbf{x}$ be a basic feasible solution. Let $B(1), \dots, B(m)$ be the indices of the basic variables, and let $B$ be the basis matrix $\left[A_{B(1)}, \dots, A_{B(m)}\right]$. Let $\mathbf{x}_B = \left(x_{B(1)}, \dots, x_{B(m)}\right)^\intercal$. Suppose we move in a direction $\mathbf{d}$ such that $d_j = 1$, and $d_i = 0$ for all non-basic $i \neq j$, or more explicitly $i \in \{1, 2, \dots, n\} \setminus \{B(1), \dots, B(m), j\}$. This direction $\mathbf{d}$ is called the $j$th *basic direction*, since it moves in the direction of the $j$th basic variable. Note that we can write

$$\mathbf{d} = (d_{B(1)}, \dots, d_{B(m)}, 0, 0, \dots, \underbrace{1}_{j\text{th entry}}, \dots, 0, 0)$$

When we move in this direction, we want to move to a feasible point. This means that we require

$$\begin{aligned} A(\mathbf{x} + \theta\mathbf{d}) &= \mathbf{b} \\ A\mathbf{d} &= 0 \\ B\mathbf{d}_B + A_j &= 0 \\ \mathbf{d}_B &= -B^{-1} A_j \end{aligned}$$

For the positivity condition, note that

$$\mathbf{x} + \theta\mathbf{d} = (x_{B(1)} + \theta d_{B(1)}, \dots, x_{B(m)} + \theta d_{B(m)}, 0, 0, \dots, \underbrace{\theta}_{j\text{th entry}}, \dots, 0, 0)$$

For this $\mathbf{x}$ to be feasible, all $x_i$ must be non-negative. Since $x_{B(i)} > 0$, there exists a small enough $\theta$ such that $\mathbf{x} + \theta\mathbf{d} \geq 0$. Hence, the $j$th basic direction is feasible.

## 6.3 Cost of basic directions

How does the cost change when $\mathbf{x} \mapsto \mathbf{x} + \theta\mathbf{d}$ where $\mathbf{d}$ is the (feasible) $j$th basic direction? The new cost is

$$\begin{aligned} \mathbf{c}^\intercal(\mathbf{x} + \theta\mathbf{d}) &= \mathbf{c}^\intercal(\mathbf{x} + \theta(-B^{-1}A_j)) \\ &= \mathbf{c}^\intercal\mathbf{x} + \theta(c_j - \mathbf{c}_B^\intercal B^{-1} A_j) \\ &= \mathbf{c}^\intercal\mathbf{x} + \theta\overline{c_j} \end{aligned}$$

**Theorem.** Let $\mathbf{x}$ be a basic feasible solution associated with a basis matrix $B$, and let $\overline{\mathbf{c}}$ be the vector of reduced costs. Then $\mathbf{x}$ is optimal if and only if $\overline{\mathbf{c}} \geq 0$.

*Proof.* This follows from the optimality conditions given previously. □

Now, if $\overline{c_j} \geq 0$ for all $j$, then this is an optimal solution. However, if any $\overline{c_j} < 0$, then we can move in the $j$th direction and decrease the cost.

## 6.4 Moving to basic feasible solutions

Suppose $\mathbf{x}$ is a basic feasible solution. If $\overline{\mathbf{c}} \geq 0$, then this is the optimum and we can stop. If $c_j < 0$ for some $j$, then moving in the $j$th feasible direction will reduce the cost by $\theta\overline{c_j}$. The amount by which the cost decreases is proportional to $\theta$, so we should choose the largest possible value of $\theta$ while retaining feasibility. We denote this largest $\theta$ with $\theta^\star$. There are two cases:

- If $\mathbf{d} \geq 0$, then $\theta$ is unbounded since $\mathbf{x} + \theta\mathbf{d} \geq 0$ for all $\theta > 0$. Therefore the optimal cost of this problem is $-\infty$.
- If $d_i < 0$ for some $i$, then we need $x_i + \theta d_i \geq 0$, so $\theta^\star \leq -\frac{x_i}{d_i}$. This then gives

$$\theta^\star = \min_{\{i\,:\,d_i<0\}} -\frac{x_i}{d_i}$$

or equivalently,

$$\theta^\star = \min_{\{i\in\{1,\ldots,m\}\,:\,d_{B(i)<0}\}} -\frac{x_{B(i)}}{d_{B(i)}}$$

Suppose the optimal cost is bounded. Let $\ell$ be the index minimising $\theta^\star$, so

$$\theta^\star = -\frac{x_{B(\ell)}}{d_{B(\ell)}}$$

Now, let us move in this direction by this amount.

**Theorem.** Let $\mathbf{y} = \mathbf{x} + \theta^\star\mathbf{d}$. $\mathbf{y}$ is feasible, and $\mathbf{c}^\intercal\mathbf{y} < \mathbf{c}^\intercal\mathbf{x}$. Then, $\mathbf{y}$ is a basic feasible solution with basis matrix

$$\overline{B} = \begin{pmatrix} \vdots & & \vdots & \vdots & \vdots & & \vdots \\ A_{B(1)} & \cdots & A_{B(\ell-1)} & A_j & A_{B(\ell+1)} & \cdots & A_{B(m)} \\ \vdots & & \vdots & \vdots & \vdots & & \vdots \end{pmatrix}$$

*Proof.* We know that $\mathbf{y}$ has exactly $m$ nonzero entries, since $y_{B(\ell)} = 0$. We know it is feasible, hence $\mathbf{y}$ is a basic feasible solution. $j$ becomes a basic variable and $B(\ell)$ is no longer. □

## 6.5 Simplex method

**Algorithm 3:** Simplex Method

**Result:** Global minimum of $\mathbf{c}^\intercal \mathbf{x}$
start at a basic feasible solution $\mathbf{x}$ with basis matrix $B = [A_{B(1)}, \dots, A_{B(m)}]$;
**repeat**
  choose $j$ such that $\overline{c_j} < 0$;
  $\mathbf{u} \leftarrow -B^{-1}A_j$;
  **if** $\mathbf{u} \leq 0$ **then** cost is $-\infty$ so terminate algorithm;
  $\theta^\star \leftarrow \min \frac{x_{B(i)}}{u_i}$ where $i \in \{1, \dots, m\}$ and $u_i > 0$;
  $\ell \leftarrow$ an index $i$ from the step above that gives the minimal value of $\theta^\star$;
  $\mathbf{x} \leftarrow \mathbf{x} - \theta^\star \mathbf{u}$;
**until** $\overline{\mathbf{c}} \geq 0$;
since $\overline{\mathbf{c}} \geq 0$, $\mathbf{x}$ is optimal

## 6.6 Tableau implementation

The full tableau implementation of the simplex method is a convenient way of executing the simplex algorithm without excessive computation. A simplex tableau contains four values of information:

| $-\mathbf{c}_B^\intercal$ | $\overline{\mathbf{c}}$ |
|---|---|
| $B^{-1}\mathbf{b}$ | $B^{-1}A$ |

The information is essentially

| $-$cost | reduced costs |
|---|---|
| vector to generate current basic feasible solution | matrix to generate basic directions |

In a more detailed form,

| $-\mathbf{c}_B^\intercal$ | $\overline{c_1}$ | $\overline{c_2}$ | $\cdots$ | $\overline{c_n}$ |
|---|---|---|---|---|
| $x_{B(1)}$ | $\vdots$ | $\vdots$ | | $\vdots$ |
| $\vdots$ | $B^{-1}A_1$ | $B^{-1}A_2$ | $\cdots$ | $B^{-1}A_n$ |
| $x_{B(m)}$ | $\vdots$ | $\vdots$ | | $\vdots$ |

To execute the simplex algorithm using this table, use the following algorithm.

**Algorithm 4:** Simplex Method (Tableau Implementation)

**Result:** Global minimum of $\mathbf{c}^\intercal \mathbf{x}$
start at a basic feasible solution $\mathbf{x}$ with basis matrix $B = [A_{B(1)}, \dots, A_{B(m)}]$;
**repeat**
  choose $j$ such that $\overline{c_j} < 0$;
  $\mathbf{u} \leftarrow -B^{-1}A_j$;
  **if** $\mathbf{u} \leq 0$ **then** cost is $-\infty$ so terminate algorithm;
  $\theta^\star \leftarrow \min \frac{x_{B(i)}}{u_i}$ where $i \in \{1, \dots, m\}$ and $u_i > 0$;
  $\ell \leftarrow$ an index $i$ from the step above that gives the minimal value of $\theta^\star$;
  $(*)$ add to each row of the tableau a constant multiple of the $\ell$th row so that $u_\ell$ becomes 1 and all other entries of the pivot column are 0;
**until** $\overline{\mathbf{c}} \geq 0$ *(when all entries in the 0th row are non-negative)*;
since $\overline{\mathbf{c}} \geq 0$, $\mathbf{x}$ is optimal

This is just the same as the simplex method discussed before, apart from step $(*)$. No proof will be given for why this step achieves the same result as the full simplex algorithm.

**Example.** Consider the problem

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^3}{\text{minimise}} \quad & -x_1 - x_2 - x_3 \\ \text{subject to} \quad & x_1 + 2x_2 + 2x_3 \leq 10 \\ & 2x_1 + x_2 + 2x_3 \leq 10 \\ & 2x_1 + 2x_2 + x_3 \leq 20 \\ & x_1, x_2, x_3 \geq 0 \end{aligned}$$

By introducing slack variables, we can write this in standard form.

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^6}{\text{minimise}} \quad & -x_1 - x_2 - x_3 \\ \text{subject to} \quad & x_1 + 2x_2 + 2x_3 + x_4 = 10 \\ & 2x_1 + x_2 + 2x_3 + x_5 = 10 \\ & 2x_1 + 2x_2 + x_3 + x_6 = 20 \\ & x_1, x_2, x_3, x_4, x_5, x_6 \geq 0 \end{aligned}$$

Observe that $(0, 0, 0, 10, 10, 20)$ is a basic feasible solution. We will use this to initiate the simplex algorithm. The corresponding basis matrix is the $3 \times 3$ identity matrix. We construct the simplex tableau by first constructing the 0th row:

- $\mathbf{c}_B = 0$ hence $\mathbf{c}_B^\intercal \mathbf{x}_B = 0$.
- $\overline{\mathbf{c}} = \mathbf{c}$.

We construct the tableau as follows.

| 0 | −1 | −1 | −1 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|
| 10 | 1 | 2 | 2 | 1 | 0 | 0 |
| 10 | 2 | 1 | 2 | 0 | 1 | 0 |
| 20 | 2 | 2 | 1 | 0 | 0 | 1 |

$\overline{c_1} < 0$, so we will descend in the 1st basic direction. Consider $\frac{10}{1}, \frac{10}{2}, \frac{20}{2}$. The smallest is $\frac{10}{2} = 5$, so the *favourite element* is the number 2 in the 1st column and 2nd row. We want to change this column to $(0, 0, 1, 0)^\intercal$ by using row operations. Denoting the rows as $R_0, \ldots, R_3$, we want to perform the operations

$$\begin{aligned} R_0 &\mapsto R_0 + \frac{1}{2}R_2 \\ R_1 &\mapsto R_1 - \frac{1}{2}R_2 \\ R_2 &\mapsto \frac{1}{2}R_2 \\ R_3 &\mapsto R_3 - R_2 \end{aligned}$$

The tableau now looks like this.

| 5 | 0 | −0.5 | 0 | 0 | 0.5 | 0 |
|---|---|---|---|---|---|---|
| 5 | 0 | 1.5 | 1 | 1 | −0.5 | 0 |
| 5 | 1 | 0.5 | 1 | 0 | 0.5 | 0 |
| 10 | 0 | 1 | −1 | 0 | −1 | 1 |

Now, $\overline{c_2} < 0$, so we will descend in the 2nd basic direction. Consider $\frac{5}{1.5}, \frac{5}{0.5}, \frac{10}{1}$. The smallest is $\frac{5}{1.5}$, so the favourite element is the 1.5 in the 1st row and 2nd column. To make the column a one-hot vector, we perform

$$\begin{aligned} R_0 &\mapsto R_0 + \frac{1}{3}R_1 \\ R_1 &\mapsto \frac{2}{3}R_1 \\ R_2 &\mapsto R_2 - \frac{1}{3}R_1 \\ R_3 &\mapsto R_3 - \frac{2}{3}R_1 \end{aligned}$$

This yields

| | | | | | | |
|---|---|---|---|---|---|---|
| $\frac{20}{3}$ | 0 | 0 | $\frac{1}{3}$ | $\frac{1}{3}$ | $\frac{1}{3}$ | 0 |
| $\frac{10}{3}$ | 0 | 1 | $\frac{2}{3}$ | $\frac{2}{3}$ | $-\frac{3}{4}$ | 0 |
| $\frac{10}{3}$ | 1 | 0 | $\frac{2}{3}$ | $-\frac{1}{3}$ | $\frac{2}{3}$ | 0 |
| $\frac{20}{3}$ | 0 | 0 | $-\frac{5}{3}$ | $-\frac{2}{3}$ | $-\frac{2}{3}$ | 1 |

Now, the 0th row has no negative values, so we are at the optimum. The optimal cost therefore is $-\frac{20}{3}$. The solution is at $\left(\frac{10}{3}, \frac{10}{3}, 0, 0, 0, \frac{20}{3}\right)$.

# 7 Game theory

## 7.1 Zero-sum games

**Definition.** A *zero-sum two-person game* is a scenario in which two players (denoted P1 and P2) have different actions they can take:

- P1 has $m$ possible actions $\{1, 2, \dots, m\}$, and
- P2 has $n$ possible actions $\{1, 2, \dots, n\}$; such that

if P1 plays move $i$ and P2 plays move $j$, then we say P1 'wins' an amount $a_{ij}$ and P2 'loses' the same amount $a_{ij}$. The matrix of results $A$ is called the *payoff matrix*. P1 chooses a row of the matrix, and P2 chooses a column, and the intersection is the outcome of the game.

Suppose P1 plays first, and chooses row $i$. P1 knows that P2 will choose the column $j$ such that $a_{ij}$ is minimised, since that will maximise P2's winnings. In particular, if P1 picks row $i$ then they can expect to win $\min_{j\in\{1,\dots,m\}} a_{ij}$. So P1 will try to solve the problem

$$\begin{aligned} &\text{maximise} && \min_{j\in\{1,\dots,m\}} a_{ij} \\ &\text{subject to} && i \in \{1, \dots, n\} \end{aligned}$$

If P2 plays first, they will try to solve the problem

$$\begin{aligned} &\text{minimise} && \max_{i\in\{1,\dots,n\}} a_{ij} \\ &\text{subject to} && j \in \{1, \dots, m\} \end{aligned}$$

**Example.** Suppose the payoff matrix is

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

P1 chooses a row, and P2 chooses a column. If P1 plays first, they choose row 2, then P2 chooses row 1, and the payoff is 3. If P2 plays first, they choose column 1, then P1 chooses row 2, and the payoff is again 3. Since the solution is the same for both problems, this point $(2, 1)$ is called a *saddle point*. The value $a_{21} = 3$ is called the *value* of the game.

**Example.** Consider the payoff matrix

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

If P1 plays first, they choose row 1, then P2 chooses column 2, and the payoff is 2. If P2 plays first, they choose column 2, then P1 chooses row 2, and the payoff is 3. Here, both players cannot play optimally simultaneously since different outcomes will occur depending on what they think their opponent will do.

## 7.2 Mixed strategies

In a mixed strategy, the players are allowed to choose their action randomly. Such mixed strategies are employed when we do not know what our opponent will pick; for example, when both players choose their option at the same time. P1 picks action $i$ with probability $p_i$, and P2 picks action $j$ with probability $q_j$, such that $\sum p_i = \sum q_j = 1$. Now, a player's strategy is encoded as a probability vector. If P1 picks the mixed strategy $(p_1, \dots, p_m)$, then the expected reward of P1 (if P2 picks a pure strategy $j$) is

$$\sum_i a_{ij} p_i$$

The optimisation problem for P1 is

$$\begin{aligned} \text{maximise} \quad & \min_{j \in \{1,\dots,n\}} \sum_i a_{ij} p_i \\ \text{subject to} \quad & \sum p_i = 1 \\ & \mathbf{p} \geq 0 \end{aligned}$$

Equivalently, where $\mathbf{e} = (1, 1, \dots, 1)^\intercal$,

$$\begin{aligned} \text{maximise} \quad & v \\ \text{subject to} \quad & A^\intercal \mathbf{p} \geq v\mathbf{e} \\ & \mathbf{e}^\intercal \mathbf{p} = 1 \\ & \mathbf{p} \geq 0 \end{aligned}$$

This $v$ is the minimum value of $A^\intercal \mathbf{p}$. P2's optimisation problem is

$$\begin{aligned} \text{minimise} \quad & \max_{i \in \{1,\dots,m\}} \sum_i a_{ij} q_j \\ \text{subject to} \quad & \sum q_j = 1 \\ & \mathbf{q} \geq 0 \end{aligned}$$

or equivalently,

$$\begin{aligned} \text{minimise} \quad & w \\ \text{subject to} \quad & A\mathbf{q} \leq w\mathbf{e} \\ & \mathbf{e}^\intercal \mathbf{q} = 1 \\ & \mathbf{q} \geq 0 \end{aligned}$$

## 7.3 Duality of mixed strategy problems

The two problems above are duals of each other. Adding slack variables, P2's problem is

$$\begin{array}{ll} \text{minimise} & w \\ \text{subject to} & A\mathbf{q} + \mathbf{s} = w\mathbf{e} \\ & \mathbf{e}^\intercal \mathbf{q} = 1 \\ & \mathbf{q} \geq 0 \\ & \mathbf{s} \geq 0 \end{array}$$

The Lagrangian of this problem is

$$\begin{aligned} L(w, \mathbf{q}, \mathbf{s}, \boldsymbol{\lambda}_1, \lambda_2) &= w + \boldsymbol{\lambda}_1^\intercal (A\mathbf{q} + \mathbf{s} - w\mathbf{e}) - \lambda_2(\mathbf{e}^\intercal \mathbf{q} - 1) \\ &= w(1 - \boldsymbol{\lambda}_1^\intercal \mathbf{e}) + (\boldsymbol{\lambda}_1^\intercal A - \lambda_2 \mathbf{e}^\intercal)\mathbf{q} + \boldsymbol{\lambda}_1^\intercal \mathbf{s} + \lambda_2 \end{aligned}$$

Thus,

$$\boldsymbol{\Lambda} = \left\{ \boldsymbol{\lambda} : \ \boldsymbol{\lambda}_1^\intercal \mathbf{e} = 1, \boldsymbol{\lambda}_1^\intercal A - \lambda_2 \mathbf{e}^\intercal \geq 0, \boldsymbol{\lambda}_1 \geq 0 \right\}$$

When $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$,

$$\inf L = \lambda_2$$

Hence the dual is

$$\begin{array}{ll} \text{maximise} & \lambda_2 \\ \text{subject to} & \boldsymbol{\lambda}_1^\intercal \mathbf{e} = 1 \\ & \boldsymbol{\lambda}_1^\intercal A \geq \lambda_2 \mathbf{e}^\intercal \\ & \boldsymbol{\lambda}_1 \geq 0 \end{array}$$

Note that $\boldsymbol{\lambda}_1 = \mathbf{p}$ and $\lambda_2 = v$ in the above formulation of P1's problem.

**Theorem.** A strategy $\mathbf{p}$ is optimal for P1 if there exist $\mathbf{q}, v$ such that
- (primal feasibility) $A^\intercal \mathbf{p} \geq v\mathbf{e}, \mathbf{e}^\intercal \mathbf{p} = 1, \mathbf{p} \geq 0$;
- (dual feasibility) $A\mathbf{q} \leq v\mathbf{e}, \mathbf{e}^\intercal \mathbf{q} = 1, \mathbf{q} \geq 0$; and
- (complementary slackness) $v = \mathbf{p}^\intercal A\mathbf{q}$

*Proof.* $(\mathbf{p}, v)$ and $(\mathbf{q}, w)$ are optimal if

$$(A\mathbf{q} - w\mathbf{e})^\intercal \mathbf{p} = 0; \mathbf{q}^\intercal (A^\intercal \mathbf{p} - v\mathbf{e}) = 0$$

which gives

$$v = w = \mathbf{p}^\intercal A\mathbf{q}$$

□

## 7.4 Finding optimal strategies

There are a number of strategies for finding optimal strategies.

(i) We can search for saddle points in the payoff matrix. If such a saddle point is found, a pure strategy aiming for this saddle point is optimal for both players.

(ii) We can search for *dominating actions*. Suppose there exist $i, i'$ such that $a_{ij} \geq a_{i'j}$ for all $j$. Then $i$ dominates $i'$, so P1 will never play $i'$ and we can simply drop this row in the matrix. A similar technique can be used to drop columns.

(iii) If these simplification techniques are not sufficient, we can simply solve the linear program using (for instance) the simplex method.

**Example.** Suppose we have a payoff matrix

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

First, observe that there is no saddle point. Note that the first row dominates the last row, so we can simplify the payoff matrix to

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

P1's strategy is $\mathbf{p} = (p, 1-p, 0)$, and the optimisation problem is

$$\begin{aligned} \text{maximise} \quad & v \\ \text{subject to} \quad & A^{\intercal}\mathbf{p} \geq v\mathbf{e} \\ & \mathbf{e}^{\intercal}\mathbf{p} = 1 \\ & \mathbf{p} \geq 0 \end{aligned}$$

which is

$$\begin{aligned} \text{maximise} \quad & v \\ \text{subject to} \quad & 2p + 3(1-p) \geq v \\ & 3p + (1-p) \geq v \\ & 4p + \frac{1}{2}(1-p) \geq v \\ & 0 \leq p \leq 1 \end{aligned}$$

and by simplifying,

$$\begin{aligned} \text{maximise} \quad & v \\ \text{subject to} \quad & v \leq 3 - p \\ & v \leq 1 + 2p \\ & v \leq \frac{1}{2} + \frac{7}{2}p \\ & 0 \leq p \leq 1 \end{aligned}$$

We can solve this graphically since it is a one-dimensional problem, or use the simplex method. We arrive at the solution $\mathbf{p} = \left(\frac{2}{3}, \frac{1}{3}, 0\right)$, i.e. $p = \frac{2}{3}$. The payoff is $\frac{7}{3}$. Player 2 has the dual optimisation problem, so we can use complementary slackness to compute P2's strategy. The first two constraints are tight, but the final constraint may not be (since it is zero in P1's strategy). Therefore $q_3 = 0$, and P2's strategy is $\mathbf{q} = (q, 1-q, 0)$. Since the value of the game is $\frac{7}{3}$, we have

$$\frac{7}{3} = \mathbf{p}^{\intercal} A \mathbf{q}$$

which lets us find $q$. Alternatively, we can use complementary slackness. Since $p_1, p_2 > 0$, the first two constraints in the dual problem must be tight.

$$2q + 3(1-q) = \frac{7}{3} \implies q = \frac{2}{3}$$

# 8 Network flows

## 8.1 Minimum cost flow

**Definition.** A *directed graph* (also known as a *digraph*) $G$ consists of a set of vertices and a set of edges; $G = (V, E)$. The edges are such that $E \subseteq V \times V$. Each edge $(i, j)$ can be thought of as an edge pointing from vertex $i$ to vertex $j$. When $E$ is symmetric (that is, $(i, j) \in E \iff (j, i) \in E$), we call $G$ an *undirected graph*.

**Definition.** Given a graph $G = (V, E)$ on $n$ vertices, we associate to every $(i, j) \in E$ the number $x_{ij}$. This represents the flow of a quantity from vertex $i$ to vertex $j$. The collection $x$ of $x_{ij}$ is called the *flow*. The flow $x$ is affected by

(i) A vector $\mathbf{b} \in \mathbb{R}^n$, where $b_i$ is the amount of flow entering vertex $i$ from outside the graph. If $b_i > 0$, then vertex $i$ is called a *source*. If $b_i < 0$, then vertex $i$ is called a *sink*.

(ii) The cost matrix $c \in \mathbb{R}^{n \times n}$, which gives the cost $c_{ij}$ per unit of flow on $(i, j) \in E$. If the flow along $(i, j)$ is $x_{ij}$, the cost for this flow is $c_{ij}x_{ij}$ (without the summation convention).

(iii) The lower bound matrix $\underline{M}$ and the upper bound matrix $\overline{M}$, which give lower and upper bounds on $x_{ij}$. In particular, for all $(i, j) \in E$, we require $\underline{m}_{ij} \leq x_{ij} \leq \overline{m}_{ij}$.

**Definition.** The *minimum cost flow* is the linear program

$$\begin{aligned} \text{minimise} \quad & \sum_{(i,j)\in E} c_{ij}x_{ij} \\ \text{subject to} \quad & \underline{m}_{ij} \leq x_{ij} \leq \overline{m}_{ij} \quad \forall (i,j) \in E \\ & b_i + \sum_{(j,i)\in E} x_{ji} = \sum_{(i,j)\in E} x_{ij} \quad \forall i \in V \end{aligned}$$

The second constraint is a conservation of flow equation. The amount of flow entering and leaving the vertex must be equal. Note that in order for the problem to be feasible, $\sum b_i = 0$; since the graph has no storage capacity at any vertex, the amount of flow that enters the graph must be the amount of flow that exits. Alternatively, we could prove this by finding the sum of the conservation of flow equations for all $i$.

**Definition.** We can define the *incidence matrix* $A : \mathbb{R}^{|V| \times |E|}$. Each column of $A$ is associated with an edge $(i, j)$. We define that this column is filled with zeroes, except for $+1$ at position $i$ and $-1$ at position $j$. We can now rewrite the conservation of flow equation as

$$A\mathbf{x} = \mathbf{b}$$

## 8.2 Transport problem

The transport problem is a special case of the minimum cost flow problem. Consider $n$ suppliers, and $m$ consumers. Each supplier $i$ has some capacity $s_i$ for how much of this good they can satisfy, and each consumer $j$ has some demand $d_j$ that they want to be fulfilled. We will assume that there is exactly as much supply as demand; that is, $\sum s_i = \sum d_j$. The cost of transporting one unit of this good from supplier $i$ to consumer $j$ is $c_{ij}$. For this problem, the graph $G$ is a *bipartite graph*; it can be separated into a set of sources and a set of sinks, and the edges are only from the sources to the sinks. The optimisation problem is

$$\begin{aligned} \text{minimise} \quad & \sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij}x_{ij} \\ \text{subject to} \quad & \sum_{j=1}^{m} x_{ij} = s_i \quad \forall i \in \{1,\dots,n\} \\ & \sum_{i=1}^{m} x_{ij} = d_j \quad \forall j \in \{1,\dots,m\} \end{aligned}$$

which is a special case of the minimum flow problem.

## 8.3 Sufficiency of transport problem

**Theorem.** Every minimum cost flow problem with either finite capacities or non-negative capacities can be translated into an equivalent transport problem.

*Proof.* Consider the minimum cost flow problem on a graph $G = (V, E)$. We may assume without loss of generality that $\underline{m}_{ij} = 0$ for all $(i, j) \in E$, because we may write $x_{ij} = \underline{m}_{ij} + \tilde{x}_{ij}$ where $\tilde{x}_{ij} > 0$. Then the conservation equation becomes

$$\tilde{b}_i + \sum_{(j,i)\in E} \tilde{x}_{ji} = \sum_{(i,j)\in E} \tilde{x}_{ij}$$

where $\tilde{b}_i = \sum_{(j,i)\in E} \underline{m}_{ji} - \sum_{(i,j)\in E} \underline{m}_{ij}$. The regional constraints are now

$$0 \le \tilde{x}_{ij} \le \overline{m}_{ij} - \underline{m}_{ij}$$

We assume that $\underline{m}_{ij} \equiv 0$ from now. If all the costs are non-negative and a particular capacity is infinite, then we can replace that capacity by a large number e.g. $\sum |b_i|$, which is the maximum amount of flow that could possibly travel along this edge. This transformation does not change the optimal solution. We have now reduced to the case where all capacities are finite.

Now, for each such minimum cost flow problem, we will construct an equivalent transport problem that has the same feasible solutions and the same costs. For each vertex $i$, we create a consumer with demand $\sum_{(i,j)\in E} \overline{m}_{ik} - b_i$. For every edge $(i, j)$, we create a supplier with supply $\overline{m}_{ij}$. The total supply and the total demand are equal, since $\sum_i b_i = 0$. We now define the cost of moving from $(i, j) \to i$ is zero. We further define the cost of moving from $(i, j) \to j$ is $c_{ij}$.

Now, suppose $x_{ij}$ flows from $(i,j) \to j$. Then $\overline{m}_{ij} - x_{ij}$ flows from $(i,j) \to i$, since the total incoming and outgoing flow from $(i,j)$ must balance. Then, since the demand at $i$ is $\sum_{(i,j)\in E} \overline{m}_{ik} - b_i$, the total flow into $i$ satisfies

$$\sum_{(i,k)\in E} (\overline{m}_{ik} - x_{ik}) + \sum_{(k,i)\in E} x_{ki} = \sum_{(i,j)\in E} \overline{m}_{ik} - b_i$$

which simplifies to the conservation equation for the minimum cost flow problem. We can easily check that $0 \le x_{ij} \le \overline{m}_{ij}$. So this mapping between the minimum cost flow problem and the transport problem preserves feasibility of solutions.

It now suffices to show that the costs of the two feasible solutions for the two problems are the same; since then we will have demonstrated a mapping between the two problems. The cost in the transport problem is $\sum_{(i,j)\in E} x_{ij}c_{ij}$ since the edge from $(i,j)$ to $i$ has zero cost. This is identical to the cost in the minimum cost flow problem. □

## 8.4 Optimality conditions for transport problem

Recall that for a linear program, there are three optimality conditions: primal feasibility, dual feasibility, and complementary slackness. These have various interpretations in the context of a transport problem.

**Theorem.** If for some feasible $x$ we have dual variables $\boldsymbol{\lambda} \in \mathbb{R}^n$ (for suppliers) and $\boldsymbol{\mu} \in \mathbb{R}^m$ (for consumers), such that:
(i) $\lambda_i + \mu_j \le c_{ij} \quad \forall (i,j) \in E$; and
(ii) $(c_{ij} - (\lambda_i + \mu_j))x_{ij} = 0 \quad \forall (i,j) \in E$
then $x$ is an optimal solution.

*Proof.* The Lagrangian of the transport problem is

$$\begin{aligned} L(x, \boldsymbol{\lambda}, \boldsymbol{\mu}) &= \sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij}x_{ij} - \sum_{i=1}^{n} \lambda_i \left( \sum_{j=1}^{m} x_{ij} - s_i \right) - \sum_{j=1}^{m} \mu_j \left( \sum_{i=1}^{n} x_{ij} - d_j \right) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{m} (c_{ij} - \lambda_i - \mu_j)x_{ij} + \sum_{i=1}^{n} \lambda_i s_i + \sum_{j=1}^{m} \mu_j d_j \end{aligned}$$

$(\boldsymbol{\lambda}, \boldsymbol{\mu})$ is dual feasible if $\lambda_i + \mu_j \le c_{ij}$ for all $i, j$. We have primal feasibility, dual feasibility, and complementary slackness, so optimality holds. □

Note that if $\boldsymbol{\lambda}, \boldsymbol{\mu}$ are optimal, then $\boldsymbol{\lambda} + k, \boldsymbol{\mu} - k$ are also optimal, since $(\lambda_i + k) + (\mu_j - k) = \lambda_i + \mu_j$. So for simplicity, we can always choose $\lambda_1 = 0$. This gives $m+n-1$ remaining Lagrange multipliers.

# 9 The transport algorithm

## 9.1 Transportation tableaux

Analogously to the simplex tableaux, for the transport problem we can create transportation tableaux. This is a convenient format for storing all relevant information for the transport problem while solving it. The transportation tableau is as follows:

| | $\mu_1$ | $\mu_2$ | $\cdots$ | $\mu_m$ | |
|---|---|---|---|---|---|
| $\lambda_1$ | $\lambda_1+\mu_1$ <br> $x_{11}$ $c_{11}$ | $\lambda_1+\mu_2$ <br> $x_{12}$ $c_{12}$ | $\cdots$ | $\lambda_1+\mu_m$ <br> $x_{1m}$ $c_{1m}$ | $s_1$ |
| $\lambda_2$ | $\lambda_2+\mu_1$ <br> $x_{21}$ $c_{21}$ | $\lambda_2+\mu_2$ <br> $x_{22}$ $c_{22}$ | $\cdots$ | $\lambda_2+\mu_m$ <br> $x_{2m}$ $c_{2m}$ | $s_2$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\ddots$ | $\vdots$ | $\vdots$ |
| $\lambda_n$ | $\lambda_n+\mu_1$ <br> $x_{n1}$ $c_{n1}$ | $\lambda_n+\mu_2$ <br> $x_{n2}$ $c_{n2}$ | $\cdots$ | $\lambda_n+\mu_m$ <br> $x_{nm}$ $c_{nm}$ | $s_n$ |
| | $d_1$ | $d_2$ | $\cdots$ | $d_m$ | |

Like with the simplex method, we must begin with a basic feasible solution to construct the initial tableau. We can construct such a basic feasible solution by using the first supplier to satisfy the first consumer, then gradually using the next suppliers and consumers as we run out of supply or demand. $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ can be deduced by considering complementary slackness. That is, if $x_{ij} > 0$ then $\lambda_i + \mu_j = c_{ij}$. For instance, consider this problem with three suppliers and four consumers. The general transportation tableau would look like this:

| | $\mu_1$ | $\mu_2$ | $\mu_3$ | $\mu_4$ | |
|---|---|---|---|---|---|
| $\lambda_1$ | $\lambda_1+\mu_1$ <br> $x_{11}$ $c_{11}$ | $\lambda_1+\mu_2$ <br> $x_{12}$ $c_{12}$ | $\lambda_1+\mu_3$ <br> $x_{13}$ $c_{13}$ | $\lambda_1+\mu_4$ <br> $x_{14}$ $c_{14}$ | $s_1$ |
| $\lambda_2$ | $\lambda_2+\mu_1$ <br> $x_{21}$ $c_{21}$ | $\lambda_2+\mu_2$ <br> $x_{22}$ $c_{22}$ | $\lambda_2+\mu_3$ <br> $x_{23}$ $c_{23}$ | $\lambda_2+\mu_4$ <br> $x_{24}$ $c_{24}$ | $s_2$ |
| $\lambda_3$ | $\lambda_3+\mu_1$ <br> $x_{31}$ $c_{31}$ | $\lambda_3+\mu_2$ <br> $x_{32}$ $c_{32}$ | $\lambda_3+\mu_3$ <br> $x_{33}$ $c_{33}$ | $\lambda_3+\mu_4$ <br> $x_{34}$ $c_{34}$ | $s_3$ |
| | $d_1$ | $d_2$ | $d_3$ | $d_4$ | |

We will consider the problem given by

$$\mathbf{s} = \begin{pmatrix} 14 \\ 10 \\ 9 \end{pmatrix}; \quad \mathbf{d} = \begin{pmatrix} 12 \\ 5 \\ 8 \\ 8 \end{pmatrix}; \quad C = \begin{pmatrix} 5 & 3 & 4 & 6 \\ 2 & 7 & 4 & 1 \\ 5 & 6 & 2 & 4 \end{pmatrix}$$

A basic feasible solution is given by

$$X = \begin{pmatrix} 12 & 2 & 0 & 0 \\ 0 & 3 & 7 & 0 \\ 0 & 0 & 1 & 8 \end{pmatrix}$$

Complementary slackness gives

$$\begin{aligned}\lambda_1 + \mu_1 &= 5\\ \lambda_1 + \mu_2 &= 3\\ \lambda_2 + \mu_2 &= 7\\ \lambda_2 + \mu_3 &= 4\\ \lambda_3 + \mu_3 &= 2\\ \lambda_3 + \mu_4 &= 4\end{aligned}$$

This is a system of seven equations for six unknowns. However, since we can always set $\lambda_1 = 0$, we can reduce this to a system of six equations for six unknowns.

$$\begin{aligned}\mu_1 &= 5\\ \mu_2 &= 3\\ \lambda_2 + \mu_2 &= 7\\ \lambda_2 + \mu_3 &= 4\\ \lambda_3 + \mu_3 &= 2\\ \lambda_3 + \mu_4 &= 4\end{aligned}$$

Hence,

$$\boldsymbol{\lambda} = \begin{pmatrix}0\\4\\2\end{pmatrix}; \quad \boldsymbol{\mu} = \begin{pmatrix}5\\3\\0\\2\end{pmatrix}$$

**Theorem.** When constructing a basic feasible solution in this way, the set of edges with strictly positive flow form a connected graph with no cycles. In particular, this graph is a *spanning tree* $T$ with exactly $m + n - 1$ edges. This allows us to always construct a system of equations as above.

No proof is given.

| | 5 | 3 | 0 | 2 | |
|---|---|---|---|---|---|
| 0 | 5<br>12 \| 5 | 3<br>2 \| 3 | 0<br>0 \| 4 | 2<br>0 \| 6 | 14 |
| 4 | 9<br>0 \| 2 | 7<br>3 \| 7 | 4<br>7 \| 4 | 6<br>0 \| 1 | 10 |
| 2 | 7<br>0 \| 5 | 5<br>0 \| 6 | 2<br>1 \| 2 | 4<br>8 \| 4 | 9 |
| | 12 | 5 | 8 | 8 | |

## 9.2 Updating the transportation tableau

First, we check if $c_{ij} \geq \lambda_i + \mu_j$ for all $i, j$. If this is true, then our solution is optimal. In our example $c_{21} \geq \lambda_2 + \mu_1$, so we are not at an optimal solution. If $(i, j) \notin T$ (where $T$ is the spanning tree above,

i.e. $x_{ij} = 0$) and $c_{ij} < \lambda_i + \mu_j$, then $(i, j)$ and the edges of $T$ form a loop. We then increase $x_{ij}$ as much as possible until another flow $x_{i'j'}$ is forced to be zero. Then we update the dual variables $\boldsymbol{\lambda}, \boldsymbol{\mu}$ and repeat.

In our example, we will introduce a flow of $x_{21} = \theta$. This will change the amount of flow along some nonzero edges. Doing this will force an update $x_{11} \mapsto x_{11} - \theta$ due to constrained demand, $x_{12} \mapsto x_{12} + \theta$ due to supply, and $x_{22} \mapsto x_{22} - \theta$ due to demand. We can then increase $\theta$ to a maximum value of 3. Now,

$$x = \begin{pmatrix} 9 & 5 & 0 & 0 \\ 3 & 0 & 7 & 0 \\ 0 & 0 & 1 & 8 \end{pmatrix}$$

We now recalculate $\boldsymbol{\lambda}, \boldsymbol{\mu}$ in the same way as above, which will give

$$\boldsymbol{\lambda} = \begin{pmatrix} 0 \\ -3 \\ -5 \end{pmatrix}; \quad \boldsymbol{\mu} = \begin{pmatrix} 5 \\ 3 \\ 7 \\ 9 \end{pmatrix}$$

Reconstructing the tableau gives

| | 5 | 3 | 7 | 9 | |
|---|---|---|---|---|---|
| 0 | 5<br>9 [5] | 3<br>3 [3] | 7<br>0 [4] | 9<br>0 [6] | 14 |
| −3 | 2<br>3 [2] | 0<br>0 [7] | 4<br>7 [4] | 6<br>0 [1] | 10 |
| 5 | 0<br>0 [5] | −2<br>0 [6] | 2<br>1 [2] | 4<br>8 [4] | 9 |
| | 12 | 5 | 8 | 8 | |

Once again there is an edge where $c_{ij} < \lambda_i + \mu_j$, notably $(i, j) = (2, 4)$, with zero flow. If $x_{ij} = \theta$, then $x_{23} \mapsto x_{23} - \theta$, $x_{34} \mapsto x_{34} - \theta$, $x_{33} \mapsto x_{33} + \theta$. We can increase $\theta$ only to 7. Once again, updating the tableau gives

| | 5 | 3 | 2 | 4 | |
|---|---|---|---|---|---|
| 0 | 5<br>9 [5] | 3<br>5 [3] | 2<br>0 [4] | 4<br>0 [6] | 14 |
| −3 | 2<br>3 [2] | 0<br>0 [7] | −1<br>0 [4] | 1<br>7 [1] | 10 |
| 0 | 5<br>0 [5] | 3<br>0 [6] | 2<br>8 [2] | 4<br>1 [4] | 9 |
| | 12 | 5 | 8 | 8 | |

In this current table, all optimality conditions are satisfied. So the solution is

$$x = \begin{pmatrix} 9 & 5 & 0 & 0 \\ 3 & 0 & 0 & 7 \\ 0 & 0 & 8 & 1 \end{pmatrix}$$

# 10 Maximum flow, minimum cut

## 10.1 Introduction

Consider the problem

$$\begin{aligned}
&\text{maximise} && \delta \\
&\text{subject to} && \sum_{\{j\,:\,(i,j)\in E\}} x_{ij} - \sum_{\{j\,:\,(j,i)\in E\}} x_{ji} = 0 \text{ for all } i \neq 1, i \neq n \\
& && \sum_{\{j\,:\,(1,j)\in E\}} x_{1j} - \sum_{\{j\,:\,(j,1)\in E\}} x_{j1} = \delta \\
& && \sum_{\{j\,:\,(n,j)\in E\}} x_{nj} - \sum_{\{j\,:\,(j,n)\in E\}} x_{jn} = -\delta \\
& && 0 \leq x_{ij} \leq c_{ij} \text{ for all } (i,j) \in E
\end{aligned}$$

This is a graph where vertex 1 is a source and vertex $n$ is a sink, and $\delta$ is the flow from vertex 1 to vertex $n$. We want to maximise the total amount of flow on the graph, constrained by a certain maximum flow $c_{ij}$ on each edge.

## 10.2 Cuts and flows

**Definition.** A *cut* of a graph $G = (V, E)$ is a partition of its vertices into two sets $(S, V \setminus S)$. The *capacity* of a cut is given by

$$C(S) = \sum_{\{(i,j)\in E\,:\,i\in S, j\in V\setminus S\}} c_{ij}$$

**Theorem.** For any feasible flow $x$ with value $\delta$, then for any cut $(S, V \setminus S)$ such that $1 \in S, n \in V \setminus S$, we have

$$\delta \leq C(S)$$

*Proof.* For any sets $X, Y \subseteq V$, we define the function

$$f_x(X, Y) = \sum_{\{(i,j)\in E\,:\,i\in X, j\in Y\}} x_{ij}$$

Note that $X, Y$ need not be disjoint. Let $(S, V \setminus S)$ be a cut such that $1 \in S, n \in V \setminus S$. We have

$$\delta = \sum_{i\in S} \left( \sum_{\{j\,:\,(i,j)\in E\}} x_{ij} - \sum_{\{j\,:\,(j,i)\in E\}} x_{ji} \right)$$

since for $i = 1$ the bracket is $\delta$ and for all others it is zero. Therefore,

$$\begin{aligned}\delta &= f_x(S, V) - f_x(V, S) \\ &= f_x(S, S) + f_x(S, V \setminus S) - f_x(S, S) - f_x(V \setminus S, S) \\ &= f_x(S, V \setminus S) - \underbrace{f_x(V \setminus S, S)}_{\geq 0} \\ &\leq f_x(S, V \setminus S) \\ &\leq C(S)\end{aligned}$$

□

## 10.3 Max-flow min-cut theorem

**Theorem.** Let $\delta^\star$ be the value of the maximum flow. Then we have

$$\delta^\star = \min\{C(S) : 1 \in S, n \in V \setminus S\}$$

So the value of the maximum flow is equal to the cut of smallest capacity.

*Proof.* A *path* $v_0, v_1, \ldots, v_k$ is a sequence of vertices such that every pair of adjacent vertices is connected by an edge, either in the forward direction or in the reverse direction. A path is called an *augmenting* path if

$$\begin{aligned} x_{v_i v_{i+1}} &< c_{v_i v_{i+1}} \quad \text{for all forward edges;} \\ x_{v_i v_{i+1}} &> 0 \quad \text{for all backward edges} \end{aligned}$$

So each forward edge must have remaining capacity, and reverse edges must have some flow. This definition allows us to state that augmenting paths are actually all paths such that altering the flow on all edges in the path can increase the total flow from 1 to $n$, while keeping the amount of flow into each vertex the same (excluding the first and last vertices in the path). Therefore, an optimal flow $x$ cannot have an augmenting path from vertex 1 to vertex $n$. Now, suppose $x$ is optimal. We define a cut:

$$S = \{1\} \cup \{i : \exists \text{ an augmenting path } 1 \to i\}$$

Therefore $n \in V \setminus S$, since there is no augmenting path from 1 to $n$. Then,

$$\delta^\star = f_x(S, V \setminus S) - f_x(V \setminus S, S)$$

But we can show that $f_x(V \setminus S, S) = 0$, so

$$\delta^\star = f_x(S, V \setminus S) = C(S)$$

as required. □

## 10.4 Ford–Fulkerson algorithm

The above proof provides a convenient method for finding an optimal flow.

**Algorithm 5:** Ford–Fulkerson Algorithm

**Result:** Optimal flow $x$
start with a feasible flow, such as $x = 0$;
**repeat**
  choose an augmenting path from 1 to $n$, and increase the flow along this path as much as possible;
**until** *no augmenting paths from* 1 *to* $n$;

**Example.** *Note that typically such graphs are represented pictorially, but due to difficulty of typesetting abstract diagrams, a matrix is substituted here.* Consider a graph given by the capacity matrix

$C =$

| $c_{ij}$ | 1 | $a$ | $b$ | $c$ | $d$ | $n$ |
|---|---|---|---|---|---|---|
| 1 | | 5 | | 5 | | |
| $a$ | | | 1 | | 4 | |
| $b$ | | | | | | 5 |
| $c$ | | | | | 2 | |
| $d$ | | | | | | 5 |
| $n$ | | | | | | |

First consider the feasible flow of $x = 0$. There exists an augmenting path $1, a, b, n$. We increase the flow by 1 in all edges, saturating edge $(a, b)$, giving the flow matrix

$x =$

| $x_{ij}$ | 1 | $a$ | $b$ | $c$ | $d$ | $n$ |
|---|---|---|---|---|---|---|
| 1 | | 1 | | | | |
| $a$ | | | 1 | | | |
| $b$ | | | | | | 1 |
| $c$ | | | | | | |
| $d$ | | | | | | |
| $n$ | | | | | | |

The path $1, a, d, n$ is now augmenting. We can increase the flow by 4 to saturate the edge $(a, d)$:

$x =$

| $x_{ij}$ | 1 | $a$ | $b$ | $c$ | $d$ | $n$ |
|---|---|---|---|---|---|---|
| 1 | | 5 | | | | |
| $a$ | | | 1 | | 4 | |
| $b$ | | | | | | 1 |
| $c$ | | | | | | |
| $d$ | | | | | | 4 |
| $n$ | | | | | | |

The path $1, c, d, n$ is augmenting. Increasing by 1,

$x =$

| $x_{ij}$ | 1 | $a$ | $b$ | $c$ | $d$ | $n$ |
|---|---|---|---|---|---|---|
| 1 | | 5 | | 1 | | |
| $a$ | | | 1 | | 4 | |
| $b$ | | | | | | 1 |
| $c$ | | | | | 1 | |
| $d$ | | | | | | 5 |
| $n$ | | | | | | |

There are no augmenting paths. We can also check that the cut $(\{1\}, \{a, b, c, d, n\})$ gives the capacity 6, equivalent to the value at $n$ so this must be optimal. We now have $\delta^\star = 6$.

**Example.** Consider a graph given by the capacity matrix

$C =$

| $c_{ij}$ | 1 | $a$ | $b$ | $c$ | $d$ | $n$ |
|---|---|---|---|---|---|---|
| 1 | | 10 | | 10 | | |
| $a$ | | | 4 | 2 | 8 | |
| $b$ | | | | | | 10 |
| $c$ | | | | | 9 | |
| $d$ | | | 6 | | | 10 |
| $n$ | | | | | | |

The path $1, a, d, n$ is augmenting. We can increase the (currently zero) flow by 8.

$x =$

| $x_{ij}$ | 1 | $a$ | $b$ | $c$ | $d$ | $n$ |
|---|---|---|---|---|---|---|
| 1 | | 8 | | | | |
| $a$ | | | | | 8 | |
| $b$ | | | | | | |
| $c$ | | | | | | |
| $d$ | | | | | | 8 |
| $n$ | | | | | | |

The path $1, c, d, n$ is also augmenting. We increase the flow by 2.

$x =$

| $x_{ij}$ | 1 | $a$ | $b$ | $c$ | $d$ | $n$ |
|---|---|---|---|---|---|---|
| 1 | | 8 | | 2 | | |
| $a$ | | | | | 8 | |
| $b$ | | | | | | |
| $c$ | | | | | 2 | |
| $d$ | | | | | | 10 |
| $n$ | | | | | | |

Now, the path $1, c, d, a, b, n$ is augmenting. $(b, a)$ here is a reverse edge. Here, we can increase the flow by 4. This will decrease the $(a, b)$ by 4.

$x =$

| $x_{ij}$ | 1 | $a$ | $b$ | $c$ | $d$ | $n$ |
|---|---|---|---|---|---|---|
| 1 | | 8 | | 6 | | |
| $a$ | | | 4 | | 4 | |
| $b$ | | | | | | 4 |
| $c$ | | | | | 6 | |
| $d$ | | | | | | 10 |
| $n$ | | | | | | |

The path $1, a, d, b, n$ is augmenting, with all forward edges. Increasing by 2,

$x =$

| $x_{ij}$ | 1 | $a$ | $b$ | $c$ | $d$ | $n$ |
|---|---|---|---|---|---|---|
| 1 | | 10 | | 6 | | |
| $a$ | | | 4 | | 6 | |
| $b$ | | | | | | 6 |
| $c$ | | | | | 6 | |
| $d$ | | | 2 | | | 10 |
| $n$ | | | | | | |

Finally, $1, c, d, b, n$ is augmenting, with all forward edges. Increasing by 3,

$$x = \begin{array}{c|cccccc} x_{ij} & 1 & a & b & c & d & n \\ \hline 1 & & 10 & & 9 & & \\ a & & & 4 & & 6 & \\ b & & & & & & 9 \\ c & & & & & 9 & \\ d & & & 5 & & & 10 \\ n & & & & & & \end{array}$$

The flow $\delta$ is now 19. The cut given by $\{1, c\}$ has capacity 19, so we are at the optimum.

## 10.5 Termination of Ford–Fulkerson

If all capacities are integers, then the algorithm will always find the optimal flow. The same argument can be used for rational numbers. At each step, the flow increases by a positive integer value, so after a finite amount of steps it will stop, as the maximum flow is bounded.

## 10.6 Bipartite matching problem

A $k$-regular bipartite graph is a graph with $\frac{n}{2}$ vertices on the left and $\frac{n}{2}$ vertices on the right, where each vertex on both the left and right has exactly $k$ edges. Suppose we want to match up the vertices on the left and right, such that each pair $(a, b)$ is joined with an edge that already exists in this graph.

**Theorem.** Every $k$-regular bipartite graph has a perfect matching.

*Proof.* First, we construct a new graph with extra vertices $1, n$. We construct edges from vertex 1 to all vertices $a$ on the left, with capacity 1. We then construct edges from all vertices $b$ on the right to vertex $n$, also with capacity 1. The original edges in the graph will be given capacity $\infty$. Then by using the cut given by 1, $\delta^* < \frac{n}{2}$. We can easily achieve the value $\delta^*$ by attaching a flow $\frac{1}{k}$ to every edge from the left to the right, and of course sending a flow of 1 along each new edge. So the maximum flow for this new graph is $\frac{n}{2}$.

Now, we know that the Ford–Fulkerson algorithm will terminate, when given the initial flow $x = 0$. But with this algorithm, all edge weights are always integers, since all capacities are integral or infinite. The only way for all edge weights to be integer values are when we have a perfect matching. So this algorithm will generate a perfect matching. □